

Лабораторная работа 3. Программные средства консолидации данных из различных источников с использованием Python и Apache Airflow

Цель: научиться работать с Apache Airflow для автоматизации процессов ETL (Extract, Transform, Load). На практике освоить настройку и выполнение DAG в Airflow для извлечения данных из различных форматов (CSV, Excel, JSON), их обработки на Python, загрузки в базу данных SQLite и отправки уведомлений по электронной почте.

Оборудование и ПО:

- Ubuntu (с Docker)
- Apache Airflow
- SQLite
- Python
- Docker
- email сервис (например, Gmail или локальный SMTP-сервер)

Исходные данные:

- Набор файлов CSV, Excel и JSON, содержащих данные для обработки.
- Конфигурация email для отправки уведомлений.

Задание по варианту 23:

Настроить DAG для периодической загрузки данных из CSV, Excel и JSON файлов в базу данных, объединения данных в единый DataFrame и последующей визуализации данных в виде диаграмм, которые сохраняются в виде изображений.

Выполнил: Петров Евгений С., БД-231м.

Ход работы

1. Развернуть VM ubuntu_mgpu.ova в VirtualBox.



2. Клонировать на ПК задание Бизнес кейс Umbrella в домашний каталог VM.

PROBLEMS TERMINAL ... bash + v [] [] ... ^ X

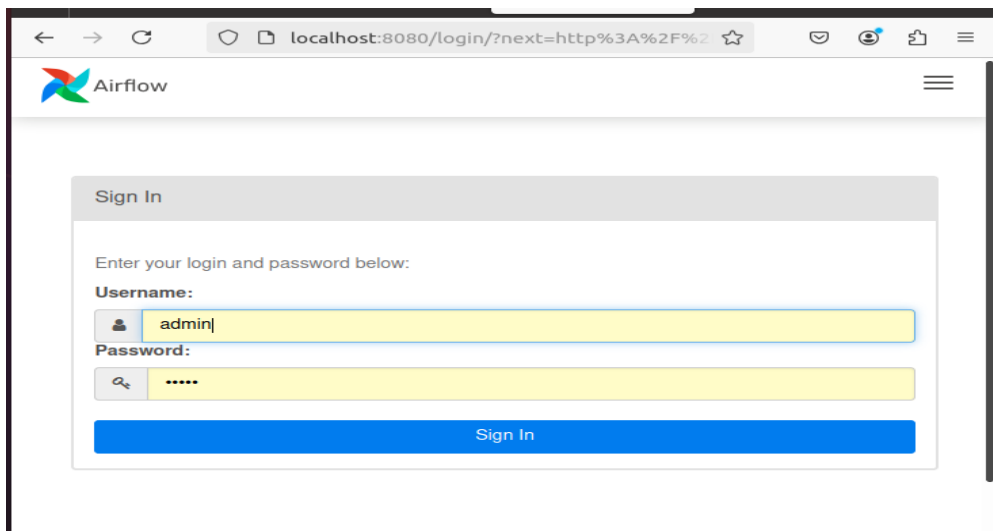
- **mgpu@mgpu-VirtualBox:~\$** cd Downloads/
- **mgpu@mgpu-VirtualBox:~/Downloads\$** git clone https://github.com/BosenkoTM/DCCAS.git
Cloning into 'DCCAS'...
remote: Enumerating objects: 23, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 23 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (23/23), 8.18 KiB | 931.00 KiB/s

PROBLEMS OUTPUT TERMINAL ... sudo-business_case_umbrella + v [] [] .

- **mgpu@mgpu-VirtualBox:~/Downloads/DCCAS\$** cd business_case_umbrella/
mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/business_case_umbrella\$ ls
dags docker-compose.yml README.md
mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/busine

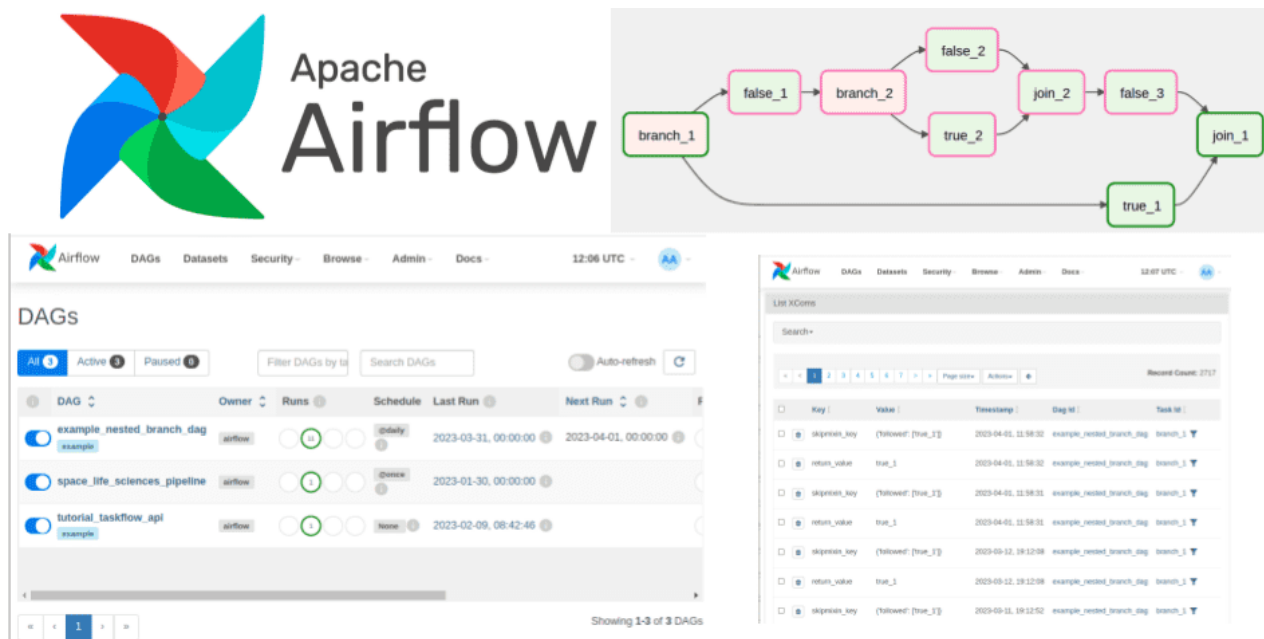
3. Запустить контейнер с кейсом.

```
mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/business_case_umbrella$ sudo docker compose up -d
[+] Running 4/5
  ⚙ Network business_case_umbrella_default      Created           3.4s
  ✓ Container business_case_umbrella-postgres-1 Started          1.3s
  ✓ Container business_case_umbrella-init-1     Started          2.5s
  ✓ Container business_case_umbrella-scheduler-1 Started          3.0s
  ✓ Container business_case_umbrella-webserver-1 Started          3.0s
mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/business_case_umbrella$
```



4. Изучить и описать основные элементы интерфейса Apache Airflow.

Основные элементы интерфейса **Apache Airflow**, представленные на скриншоте:



1. DAGS

DAGS (Directed Acyclic Graphs) – это основной элемент в Airflow, представляющий собой структуру, описывающую последовательность выполнения задач. В интерфейсе вы можете:

- Видеть список доступных DAGS.
- Проверять статус каждого DAG (активен, приостановлен и т.д.).
- Открывать визуализацию графа DAG для просмотра зависимостей между задачами.
- Запускать задачи вручную и просматривать их логи.

2. Datasets

Datasets представляют собой данные, которые потребляются или создаются в процессе выполнения задач. В интерфейсе Airflow можно видеть, какие датасеты используются в DAG, их состояния и метаданные. Это помогает в управлении зависимостями между задачами и в гарантии, что данные используются корректно.

3. Security

Элемент безопасности позволяет управлять доступом к интерфейсу Airflow. Здесь можно:

- Настраивать роли и разрешения для пользователей.
- Ограничивать доступ к определенным DAGS или функционалу на основе ролей.

4. Browse

Этот элемент интерфейса предоставляет доступ к различным данным и информации:

- Вы можете просматривать выполненные экземпляры DAG и их задачи (task instances).
- Доступ к информации о привязках к метаданным (например, логам и результатам выполнения).

- Возможность просматривать состояния выполнения задач.

5. Admin

Раздел администрирования предоставляет инструменты для управления настройками Airflow:

- Управление пользователями и ролями.
- Настройки подключения к базам данных и другим системам.
- Настройка параметров выполнения и конфигурации.

6. Docs

Этот раздел содержит документацию по Airflow, включая:

- Описание доступных операторов и как их использовать.
- Лучшие практики для создания и управления DAG.
- Примеры и справочные материалы.

7. Status: All, Active, Paused

В интерфейсе Airflow вы можете фильтровать DAGs по их статусу:

- ****All****: Показывает все DAGs.
- ****Active****: Показывает только активные (включенные) DAGs.
- ****Paused****: Показывает приостановленные DAGs.

8. Autorefresh

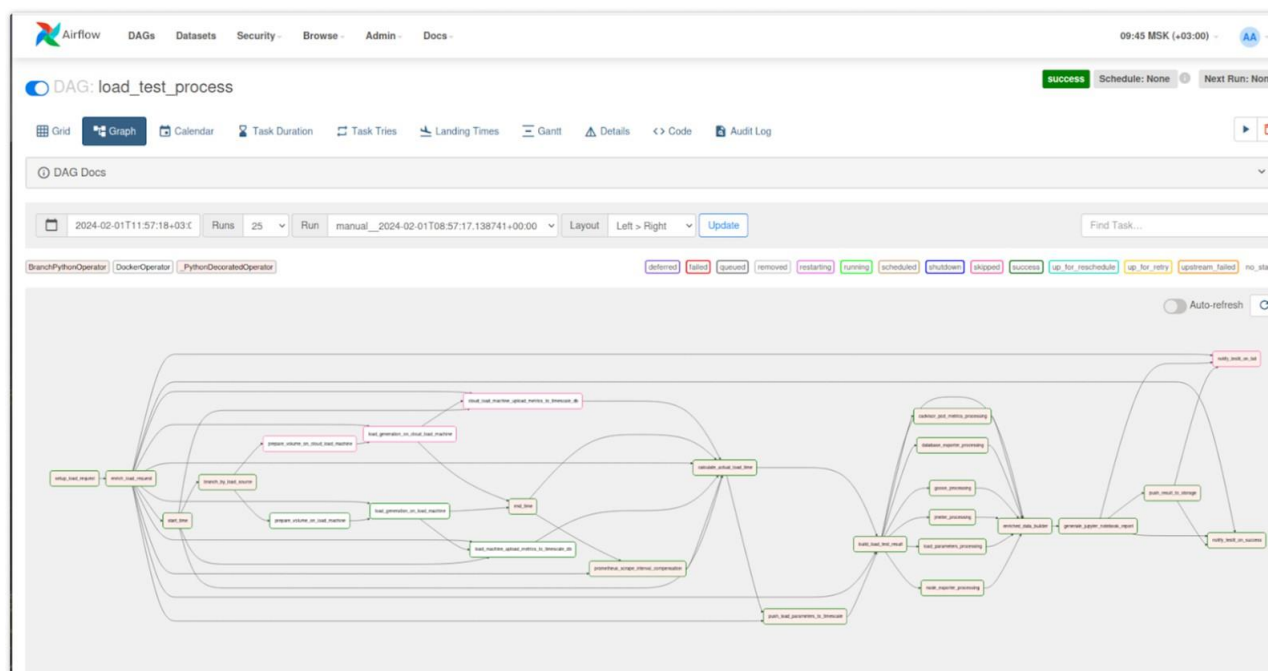
Функция автоматического обновления значения в реальном времени — это удобный инструмент, который позволяет пользователям следить за состоянием задачи и графов без необходимости вручную обновлять страницу.

9. Filter

Элемент фильтрации позволяет пользователям быстро находить конкретные DAGs или задачи по имени, статусу или другим критериям. Это особенно полезно в больших проектах с множеством DAGs.

Эти элементы интерфейса в Apache Airflow помогают пользователям эффективно управлять и мониторить их рабочие процессы, обеспечивая простоту и удобство использования.

На скриншоте представлен интерфейс графического представления **DAG** в Apache Airflow:



Apache Airflow предоставляет мощный интерфейс для графического представления Directed Acyclic Graphs (DAG), что позволяет пользователям эффективно управлять и мониторить задачи. Давайте рассмотрим основные элементы интерфейса:

1. Grid (Сетка):

Это одно из представлений DAG, где отображаются все задачи графа. Каждая ячейка в сетке соответствует конкретной задаче, и пользователи могут видеть состояние задач, такие как «завершено», «в процессе» или «неудачно».

2. Graph (Граф):

Это визуальное представление зависимостей между задачами. Задачи отображаются в виде узлов, а зависимости между ними - в виде рёбер. Это позволяет пользователям легко понять, как задачи связаны друг с другом и в каком порядке они должны выполняться.

3. Calendar (Календарь):

Этот элемент интерфейса позволяет пользователю просматривать, когда были выполнены конкретные задачи и каков их граф выполнения. Календарь может включать отметки для успешных и неудачных запусков задач.

4. Task Duration (Длительность задач):

Этот параметр показывает, сколько времени понадобилось для выполнения каждой задачи. Это может помочь в определении узких мест в процессе и оптимизации производительности.

5. Task Tries (Попытки задач):

В этом разделе отображается количество попыток выполнения каждой задачи. Airflow автоматически повторяет выполнение задач в случае неудачи, и этот элемент позволяет пользователям увидеть, сколько раз каждая задача пыталась выполниться.

6. Landing Time (Время завершения):

Этот элемент показывает, когда была завершена ссылка (проверка) задач. Это может быть важно для анализа временных задержек и повышения эффективности обработки данных.

7. Gantt (График Ганта):

Этот элемент, который показывает выполнение задач по временной шкале. Пользователи могут видеть, какие задачи выполняются параллельно, а также их взаимные зависимости, что позволяет оценить общую эффективность DAG.

8. Details (Детали):

Этот раздел предоставляет пользователям детальную информацию о конкретной задаче, включая ее состояние, параметры, лог выполнения и любые другие метаданные, которые могут быть важны для диагностики или мониторинга.

9. Code (Код):

В этом разделе можно просмотреть исходный код DAG. Это полезно для разработчиков, чтобы быстро просмотреть настройки и логику выполнения.

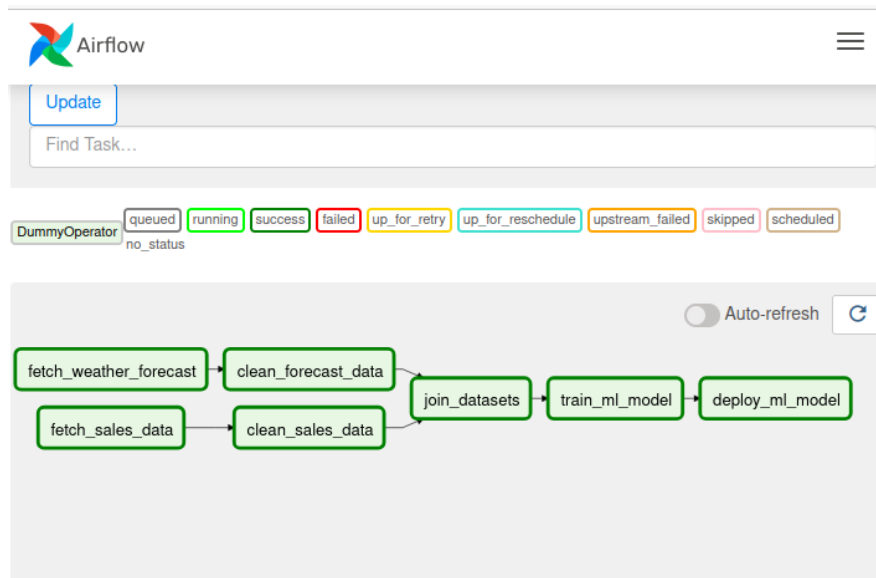
10. Audit Log (Журнал аудита):

Этот элемент сохраняет записи обо всех изменениях и событиях, происходящих в процессе выполнения DAG. Это важно для мониторинга, анализа и обеспечения безопасности.

11. Auto-refresh (Автообновление):

Эта функция автоматически обновляет интерфейс через определенные промежутки времени, что позволяет пользователям оставаться в курсе текущего состояния задач и DAG без необходимости ручного обновления страницы

На скриншоте представлен интерфейс графического представления **Graph View** в Apache Airflow:



Графический интерфейс представляет собой один из основных способов визуализации рабочих процессов (DAG - Directed Acyclic Graph) в Apache Airflow. Он предоставляет пользователям интуитивно понятный способ отслеживания состояния задач и управления ими. Вот основные элементы интерфейса Graph View:

1. Узлы (Nodes)

Каждый узел в графе представляет собой конкретную задачу (task) в рамках рабочего процесса. Узлы могут отображать различную информацию:

- ****Имя задачи****: Отображается в узле.
- ****Состояние задачи****: Цвет узла указывает на текущее состояние задачи.

2. Цвета узлов

Цвета узлов помогают быстро оценивать состояние задач:

- ****Синий****: Задача ожидает выполнения.
- ****Зеленый****: Задача успешно выполнена и завершена.
- ****Красный****: Задача завершилась с ошибкой (неудача).

- ****Серый****: Задача не была запущена, например, если она отключена.
- ****Оранжевый****: Задача находится в состоянии "выполняется" или "обрабатывается".
- ****Желтый****: Задача в состоянии "пауза" или "ожидание".

3. Связи (Dependencies)

Стрелки между узлами показывают зависимости между задачами. Это помогает понять, в каком порядке задачи должны быть выполнены. Зависимости могут быть:

- ****Последовательные****: Одна задача должна завершиться, чтобы началась следующая.
- ****Параллельные****: Несколько задач могут выполняться одновременно.

4. Параметры фильтрации

В графическом интерфейсе Airflow можно переключаться между различными видами представления (например, "Graph View", "Tree View" и "Gantt View"). Также есть возможность фильтровать задачи по статусу, дате выполнения и другим параметрам.

5. Всплывающие подсказки

Наведение курсора на узел задачи для вызова всплывающей подсказки, которая может содержать дополнительную информацию, такую как время выполнения, продолжительность, и другие метрики.

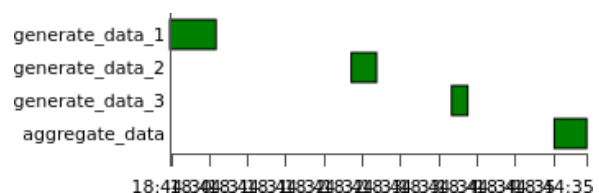
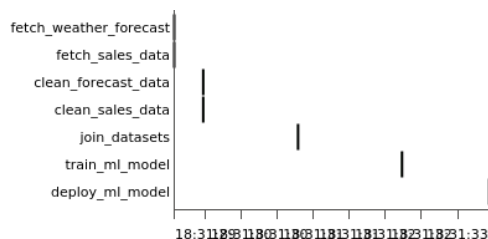
6. Панель инструментов

В верхней части интерфейса находится панель инструментов, позволяющая:

- Перезапустить DAG.
- Задавать параметры выполнения (например, выбрать дату).
- Получить доступ к журналам выполнения задач.

7. Состояние DAG

В верхней части страницы тоже может отображаться общее состояние DAG, включая общее количество задач, количество успешных и завершившихся с ошибками задач. На скриншоте представлен интерфейс графического представления **Gantt** в Apache Airflow, которая позволяет отслеживать временные интервалы выполнения каждой задачи:



Основные особенности этой вкладки:

1. Диаграмма Ганта:

- **Оси времени**: горизонтальная ось представляет собой временную шкалу, которая показывает время выполнения задач. Она позволяет визуально оценить, как долго выполнялась каждая задача и в какие моменты времени они начинались и заканчивались.
 - **Задачи**: Каждая задача представлена горизонтальной полосой. Начало полосы соответствует времени старта задачи, а её длина указывает на продолжительность выполнения.
2. **Цветовые индикаторы**: подобно другим видам в Airflow, цвет полос задач на диаграмме Ганта показывает их статус.
 3. **Интерактивность**: нажав на любую полосу на диаграмме Ганта, можно получить более детальную информацию о задаче.

5. Выполнить индивидуальное задание.

Задание 23: Настроить DAG для периодической загрузки данных из CSV, Excel и JSON файлов в базу данных, объединения данных в единый DataFrame и последующей визуализации данных в виде диаграмм, которые сохраняются в виде изображений.

1. Создание файла DAG

1. Файл CSV: данные о сотрудниках и их квалификациях (имя, квалификация, стаж).
2. Файл Excel: данные о проектах (проект, сотрудник, часы работы).
3. Файл JSON: данные о стоимости часа работы в зависимости от квалификации сотрудника.

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from datetime import datetime
import pandas as pd
import random
import json
import matplotlib.pyplot as plt
import os
from sqlalchemy import create_engine

# Генерация CSV файла с данными о сотрудниках
def generate_employee_csv():
    data_csv = {
        'Имя': [f'Сотрудник {i}' for i in range(1, 101)],
        'Квалификация': [random.choice(['Junior', 'Middle', 'Senior']) for _ in range(100)],
        'Стаж': [random.randint(1, 15) for _ in range(100)]
    }
    df_csv = pd.DataFrame(data_csv)
    df_csv.to_csv('employees_data.csv', index=False, encoding='utf-8')

# Генерация Excel файла с данными о проектах
def generate_project_excel():
    projects = [f'Проект {j}' for j in range(1, 21)]
    data_excel = {
        'Проект': [],
        'Сотрудник': [],
        'Часы работы': []
    }

    for project in projects:
        for _ in range(5): # Каждый проект включает 5 сотрудников
            data_excel['Проект'].append(project)
            data_excel['Сотрудник'].append(random.choice([f'Сотрудник {i}' for i in range(1, 101)]))
            data_excel['Часы работы'].append(random.randint(1, 10))

    df_excel = pd.DataFrame(data_excel)
    df_excel.to_excel('projects_data.xlsx', index=False)

# Генерация JSON файла с данными о стоимости часа работы
def generate_hourly_rate_json():
    data_json = {
        'Квалификация': ['Junior', 'Middle', 'Senior'],
        'Стоимость часа': [10, 20, 30]
    }

    with open('hourly_rate_data.json', 'w', encoding='utf-8') as f:
        json.dump(data_json, f, ensure_ascii=False)
```

```

# Загрузка данных в базу данных
def load_data_to_db():
    # Загрузка данных
    df_csv = pd.read_csv('employees_data.csv', encoding='utf-8')
    df_excel = pd.read_excel('projects_data.xlsx')
    with open('hourly_rate_data.json', 'r', encoding='utf-8') as f:
        data_json = json.load(f)
    df_json = pd.DataFrame(data_json)

    # Объединение данных
    merged_df = pd.merge(df_excel, df_csv, on='Сотрудник', how='left')
    merged_df = pd.merge(merged_df, df_json, left_on='Квалификация', right_on='Квалификация',
        how='left')

    # Создание соединения с базой данных
    engine = create_engine('sqlite:///employees_projects.db') # Используем SQLite
    merged_df.to_sql('employee_project_data', con=engine, index=False, if_exists='replace')

    # Визуализация данных
    def visualize_data():
        # Загрузка данных из базы
        engine = create_engine('sqlite:///employees_projects.db')
        df = pd.read_sql('SELECT * FROM employee_project_data', con=engine)

        # Группировка данных
        df_grouped = df.groupby('Квалификация').agg({'Часы работы': 'sum'}).reset_index()

        # Визуализация
        plt.figure(figsize=(10, 6))
        plt.bar(df_grouped['Квалификация'], df_grouped['Часы работы'], color='blue')
        plt.title('Общее количество часов работы по квалификации')
        plt.xlabel('Квалификация')
        plt.ylabel('Часы работы')
        plt.savefig('hours_by_qualification.png')
        plt.close()

    # Определение DAG
    dag = DAG('employee_project_data_dag',
        description='Загрузка данных сотрудников и проектов в базу данных и их визуализация',
        schedule_interval='@daily',
        start_date=datetime(2023, 10, 1),
        catchup=False)

    # Определение задач
    task_generate_employee_csv = PythonOperator(
        task_id='generate_employee_csv',
        python_callable=generate_employee_csv,
        dag=dag
    )

    task_generate_project_excel = PythonOperator(
        task_id='generate_project_excel',
        python_callable=generate_project_excel,
        dag=dag
    )

```

```
task_generate_hourly_rate_json = PythonOperator(  
task_id='generate_hourly_rate_json',  
python_callable=generate_hourly_rate_json,  
dag=dag  
)
```

```
task_load_data_to_db = PythonOperator(  
task_id='load_data_to_db',  
python_callable=load_data_to_db,  
dag=dag  
)
```

```
task_visualize_data = PythonOperator(  
task_id='visualize_data',  
python_callable=visualize_data,  
dag=dag  
)
```

2. Запуск DAG:

Запустилось все с 3 раза. До этого были ошибки, ниже пришлось установить дополнительные модули.

The screenshot displays the Apache Airflow web interface in a Firefox browser. The URL is `localhost:8080/tree?dag_id=employee_project_data_dag`. The interface shows the DAG 'employee_project_data_dag' with the description 'Загрузка данных сотрудников и проектов в базу данных и их визуализация'. The 'Tree View' is selected, showing a DAG with five tasks: 'generate_employee_csv', 'generate_project_excel', 'generate_hourly_rate_json', 'load_data_to_db', and 'visualize_data'. A legend at the top right indicates the status of tasks: queued (grey), running (green), success (green), failed (red), up_for_retry (yellow), up_for_reschedule (cyan), upstream_failed (orange), skipped (pink), scheduled (brown), and no_status (white). The bottom of the interface shows the version 'v2.0.0' and the Git commit hash 'ab5f770bfc8c690cbe4d0825896325aca0beeca'.

3. Предварительно перед этим был установлен модуль openpyxl, matplotlib, sqlalchemy, apache-airflow[sqlite], pillow/numpy:

```
Activities Terminal OKT 19 15:05
airflow@a20fd9853b27: /opt/airflow

mpgw@mpgw-VirtualBox: ~/Downloads/DCCAS/business_case_umbrella$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED        STATUS        PORTS                               NAMES
9c936dd5d59e       apache/airflow:2.0.0-python3.8 /usr/bin/dumb-init -r   39 minutes ago Up 39 minutes    0.0.0.0:8080->8080/tcp, :::8080->8080/tcp    business_case_umbrella-websvr-1
a20fd9853b27       apache/airflow:2.0.0-python3.8 /usr/bin/dumb-init -r   39 minutes ago Up 38 minutes    8080/tcp                                     business_case_umbrella-scheduler-1
ae396ec037b        postgres:12-alpine  docker-entrypoint.sh     39 minutes ago Up 39 minutes    0.0.0.0:5432->5432/tcp, :::5432->5432/tcp    business_case_umbrella-postgres-1

mpgw@mpgw-VirtualBox: ~/Downloads/DCCAS/business_case_umbrella$ sudo docker exec -it a20fd9853b27
See 'docker exec --help'.

Usage: docker exec [OPTIONS] CONTAINER COMMAND [ARG...]

Execute a command in a running container

mpgw@mpgw-VirtualBox: ~/Downloads/DCCAS/business_case_umbrella$ sudo docker exec -it a20fd9853b27 bash
airflow@a20fd9853b27: /opt/airflow$ pip install openpyxl
Defaulting to user installation because normal site-packages is not writeable
Collecting openpyxl
  Downloading openpyxl-3.1.5-py3-none-any.whl (250 kB)
    | 250 kB 988 kB/s
Collecting et-xmlfile
  Downloading et_xmlfile-1.1.0-py3-none-any.whl (4.7 kB)
Installing collected packages: et-xmlfile, openpyxl
Successfully installed et-xmlfile-1.1.0 openpyxl-3.1.5
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
airflow@a20fd9853b27: /opt/airflow$ pip install matplotlib
Defaulting to user installation because normal site-packages is not writeable
Collecting matplotlib
  Downloading matplotlib-3.7.5-cp38-cp38-manylinux_2_12_x86_64.manylinux2010_x86_64.whl (9.2 MB)
    | 9.2 MB 280 kB/s
Requirement already satisfied: pyparsing>=2.3.1 in /home/airflow/.local/lib/python3.8/site-packages (from matplotlib) (2.4.7)
Collecting pillow<=6.2.0
  Downloading pillow-10.4.0-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.4 MB)
    | 4.4 MB 2.3 MB/s
Collecting importlib-resources>=3.2.0; python_version < "3.10"
  Downloading importlib_resources-6.4.5-py3-none-any.whl (26 kB)
Requirement already satisfied: python-dateutil>=2.7 in /home/airflow/.local/lib/python3.8/site-packages (from matplotlib) (2.8.1)
Requirement already satisfied: packaging>=20.0 in /home/airflow/.local/lib/python3.8/site-packages (from matplotlib) (20.9)
Collecting contourpy>=1.0.1
  Downloading contourpy-1.1.1-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (301 kB)
    | 301 kB 1.4 MB/s
Requirement already satisfied: numpy>=1.20 in /home/airflow/.local/lib/python3.8/site-packages (from matplotlib) (1.20.1)
Collecting fonttools>=4.22.0
  Downloading fonttools-4.54.1-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (4.7 MB)
    | 4.7 MB 1.0 MB/s
Collecting kiwisolver>=1.0.1
  Downloading kiwisolver-1.4.7-cp38-cp38-manylinux_2_5_x86_64.manylinux1_x86_64.whl (1.2 MB)
    | 1.2 MB 5.1 MB/s
Collecting cycler>=0.10
  Downloading cycler-0.12.1-py3-none-any.whl (8.3 kB)
Requirement already satisfied: zipp>=3.1.0; python_version < "3.10" in /home/airflow/.local/lib/python3.8/site-packages (from importlib-resources>=3.2.0; python_version < "3.10"->matplotlib) (3.4.0)
Requirement already satisfied: six>=1.5 in /home/airflow/.local/lib/python3.8/site-packages (from python-dateutil>=2.7->matplotlib) (1.15.0)
Installing collected packages: pillow, importlib-resources, contourpy, fonttools, kiwisolver, cycler, matplotlib
Attempting uninstall: importlib-resources
Found existing installation: importlib-resources 1.5.0
Uninstalling importlib-resources-1.5.0:
Successfully uninstalled importlib-resources-1.5.0
```

```
Activities Terminal OKT 19 15:06
airflow@a20fd9853b27: /opt/airflow

You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
airflow@a20fd9853b27: /opt/airflow$ pip install sqlalchemy
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: sqlalchemy in /home/airflow/.local/lib/python3.8/site-packages (1.3.23)
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
airflow@a20fd9853b27: /opt/airflow$ pip install --upgrade sqlalchemy
Defaulting to user installation because normal site-packages is not writeable
Collecting sqlalchemy
  Downloading SQLAlchemy-2.0.36-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (3.1 MB)
    | 3.1 MB 989 kB/s
Collecting typing-extensions>=4.6.0
  Downloading typing_extensions-4.12.2-py3-none-any.whl (37 kB)
Requirement already satisfied: zipp>=3.1.0; python_version < "3.13" and (platform_machine == "aarch64" or (platform_machine == "ppc64le" or (platform_machine == "x86_64" or (platform_machine == "amd64" or (platform_machine == "AMD64" or (platform_machine == "win32" or platform_machine == "WIN32"))))) in /home/airflow/.local/lib/python3.8/site-packages (from sqlalchemy) (1.0.0)
Installing collected packages: typing-extensions, sqlalchemy
Attempting uninstall: typing-extensions
Found existing installation: typing-extensions 3.7.4.3
Uninstalling typing-extensions-3.7.4.3:
Successfully uninstalled typing-extensions-3.7.4.3
Attempting uninstall: sqlalchemy
Found existing installation: SQLAlchemy 1.3.23
Uninstalling SQLAlchemy-1.3.23:
Successfully uninstalled SQLAlchemy-1.3.23
ERROR: After October 2020 you may experience errors when installing or updating packages. This is because pip will change the way that it resolves dependency conflicts.

We recommend you use --use-feature=2020-resolver to test your packages with the new resolver before it becomes the default.

pip 0.2.0 requires typing-extensions<4.0.0,>=3.7.4, but you'll have typing-extensions 4.12.2 which is incompatible.
apache-airflow 2.0.0 requires importlib-resources<1.4, but you'll have importlib-resources 6.4.5 which is incompatible.
apache-airflow 2.0.0 requires sqlalchemy<2.0.0,>=1.3.10, but you'll have sqlalchemy 2.0.36 which is incompatible.
Successfully installed sqlalchemy-2.0.36 typing-extensions-4.12.2
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
airflow@a20fd9853b27: /opt/airflow$ pip install apache-airflow[sqlite]
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: apache-airflow[sqlite] in /home/airflow/.local/lib/python3.8/site-packages (2.0.0)
WARNING: apache-airflow 2.0.0 does not provide the extra 'sqlite'
Requirement already satisfied: dill<0.4,>=0.2.2 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (0.3.3)
Requirement already satisfied: rich<9.2.0 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (9.2.0)
Requirement already satisfied: tabulate<0.9,>=0.7.5 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (0.8.7)
Requirement already satisfied: cached-property<=1.5 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (1.5.2)
Requirement already satisfied: requests<2.24.0,>=2.20.0 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (2.23.0)
Requirement already satisfied: apache-airflow-providers-http in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (1.1.0)
Requirement already satisfied: apache-airflow-providers-ftp in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (1.0.1)
Requirement already satisfied: connexion[flask-swagger-ui]<3,>=2.6.0 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (2.7.0)
Requirement already satisfied: flask<2.0,>=1.1.0 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (1.1.2)
Requirement already satisfied: pendulum<2.0 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (2.1.2)
Requirement already satisfied: python-dateutil<3,>=2.3 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (2.8.1)
Requirement already satisfied: pygments<3.0,>=2.6.1 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (2.8.0)
Requirement already satisfied: flask-swagger==0.2.19 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (0.2.13)
Requirement already satisfied: itsdangerous<=1.1.0 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (1.1.0)
Requirement already satisfied: importlib-metadata<=1.7; python_version < "3.9" in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (1.7.0)
Requirement already satisfied: tzlocal<2.0.0,>=1.4 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (1.5.1)
Requirement already satisfied: croniter<0.4,>=0.3.17 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (0.3.37)
Requirement already satisfied: flask<2.0,>=2.0.1 in /home/airflow/.local/lib/python3.8/site-packages (from apache-airflow[sqlite]) (2.11.3)
```

```
Activities Terminal
airflow@a20fd9853b27: /opt/airflow

Downloading cycloer-0.12.1-py3-none-any.whl (8.3 kB)
Requirement already satisfied: zipp>=3.1.0; python_version < "3.10" in /home/airflow/.local/lib/python3.8/site-packages (from importlib-resources==3.2.0; python_version < "3.10"->matplotlib) (3.4.0)
Requirement already satisfied: six>=1.5 in /home/airflow/.local/lib/python3.8/site-packages (from python-dateutil==2.7->matplotlib) (1.15.0)
Installing collected packages: pillow, importlib-resources, contourpy, fonttools, ktlwsolver, cycloer, matplotlib
Attempting uninstall: importlib-resources
Found existing installation: importlib-resources 1.5.0
Uninstalling importlib-resources-1.5.0:
Successfully uninstalled importlib-resources-1.5.0
ERROR: After October 2020 you may experience errors when installing or updating packages. This is because pip will change the way that it resolves dependency conflicts.
We recommend you use --use-feature=2020-resolver to test your packages with the new resolver before it becomes the default.

apache-airflow 2.0.0 requires importlib-resources<=1.4, but you'll have importlib-resources 6.4.5 which is incompatible.
Successfully installed contourpy-1.1.1 cycloer-0.12.1 fonttools-4.54.1 importlib-resources-6.4.5 ktlwsolver-1.4.7 matplotlib-3.7.5 pillow-10.4.0
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.

airflow@a20fd9853b27: /opt/airflow$ pip install pillow
Defaulting to user installation because normal site-packages is not writeable
ERROR: Could not find a version that satisfies the requirement pillow (from versions: none)
ERROR: no matching distribution found for pillow
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.

airflow@a20fd9853b27: /opt/airflow$ pip install numpy
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in /home/airflow/.local/lib/python3.8/site-packages (1.20.1)
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.

airflow@a20fd9853b27: /opt/airflow$ pip install --upgrade pillow numpy
Usage:
  pip install [options] <requirement specifier> [package-index-options] ...
  pip install [options] -r <requirements file> [package-index-options] ...
  pip install [options] [-e] <vcs project url> ...
  pip install [options] [-e] <local project path> ...
  pip install [options] <archive url/path> ...

no such option: -u

airflow@a20fd9853b27: /opt/airflow$ pip install pillow
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pillow in /home/airflow/.local/lib/python3.8/site-packages (10.4.0)
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.

airflow@a20fd9853b27: /opt/airflow$ pip install --upgrade pillow numpy
Defaulting to user installation because normal site-packages is not writeable
Requirement already up-to-date: pillow in /home/airflow/.local/lib/python3.8/site-packages (10.4.0)
Collecting numpy
  Downloading numpy-1.24.4-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (17.3 MB)
    17.3 MB 15 KB/s
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.20.1
    Uninstalling numpy-1.20.1:
      Successfully uninstalled numpy-1.20.1
ERROR: After October 2020 you may experience errors when installing or updating packages. This is because pip will change the way that it resolves dependency conflicts.
We recommend you use --use-feature=2020-resolver to test your packages with the new resolver before it becomes the default.
```

```
Activities Terminal
airflow@a20fd9853b27: /opt/airflow

Requirement already satisfied: six>=1.5 in /home/airflow/.local/lib/python3.8/site-packages (from python-dateutil<3,=>1.2.5->flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.15.0)
Requirement already satisfied: tzpp>=0.5 in /home/airflow/.local/lib/python3.8/site-packages (from importlib-metadata==1.7; python_version < "3.9"->apache-airflow[sqllite]) (3.4.0)
Requirement already satisfied: pytz in /home/airflow/.local/lib/python3.8/site-packages (from tzlocal<2.0.0,=>1.4->apache-airflow[sqllite]) (2020.5)
Requirement already satisfied: matplotlib in /home/airflow/.local/lib/python3.8/site-packages (from contourpy<0.4,=>0.3.17->apache-airflow[sqllite]) (3.7.5)
Requirement already satisfied: ktlwsolver in /home/airflow/.local/lib/python3.8/site-packages (from ktlwsolver==1.4.7->matplotlib) (1.4.7)
Requirement already satisfied: fonttools in /home/airflow/.local/lib/python3.8/site-packages (from fonttools==4.54.1->matplotlib) (4.54.1)
Requirement already satisfied: importlib-resources==3.2.0 in /home/airflow/.local/lib/python3.8/site-packages (from importlib-resources==3.2.0; python_version < "3.10"->matplotlib) (3.4.0)
Requirement already satisfied: sqlalchemy==2.0.36 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (2.0.36)
Requirement already satisfied: flask==2.0.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (2.0.1)
Requirement already satisfied: flask-openid2==1.2.5 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.3.0)
Requirement already satisfied: marshmallow-sqlalchemy<0.24.0,=>0.22.0 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (0.23.1)
Requirement already satisfied: marshmallow==2.1.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (2.1.1)
Requirement already satisfied: sqlalchemy==2.0.36 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (2.0.36)
Requirement already satisfied: flask-jwt-extended==4.0.3 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (3.25.1)
Requirement already satisfied: flask-openid==2.0.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.5.1)
Requirement already satisfied: apspec[yaml]<4,=>3.3 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (3.3.2)
Requirement already satisfied: email-validator<2,=>1.0.5 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.1.2)
Requirement already satisfied: flask-babel==2.0.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.0.0)
Requirement already satisfied: pyjwt==1.7.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.7.1)
Requirement already satisfied: prison<0.0,=>0.1.3 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (0.1.3)
Requirement already satisfied: wtforms in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (2.3.3)
Requirement already satisfied: text-unidecode==1.3 in /home/airflow/.local/lib/python3.8/site-packages (from python-slugify==5.0,=>3.0.0->apache-airflow[sqllite]) (1.3)
Requirement already satisfied: pyyaml==5.4.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (5.4.1)
Requirement already satisfied: cffi==1.12 in /home/airflow/.local/lib/python3.8/site-packages (from cryptography==0.9.3->apache-airflow[sqllite]) (1.14.5)
Requirement already satisfied: greenlet==0.4.17; python_version >= "3" and (platform_machine == "aarch64" or (platform_machine == "ppc64le" or (platform_machine == "s390x" or (platform_machine == "x86_64" or (platform_machine == "AMD64" or (platform_machine == "win32" or platform_machine == "WIN32"))))) in /home/airflow/.local/lib/python3.8/site-packages (from sqlalchemy==2.0.36->1.3.18->apache-airflow[sqllite]) (1.0.0)
Requirement already satisfied: sqlalchemy==2.0.36 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (2.0.36)
Requirement already satisfied: flask-babel==2.0.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.0.0)
Requirement already satisfied: pycparser in /home/airflow/.local/lib/python3.8/site-packages (from cffi==1.12->cryptography==0.9.3->apache-airflow[sqllite]) (2.0)
Requirement already satisfied: flask-openid==2.0.1 in /home/airflow/.local/lib/python3.8/site-packages (from flask-appbuilder==3.1.1->apache-airflow[sqllite]) (1.5.1)
Installing collected packages: importlib-resources, sqlalchemy, typing-extensions
  Attempting uninstall: importlib-resources
    Found existing installation: importlib-resources 6.4.5
    Uninstalling importlib-resources-6.4.5:
      Successfully uninstalled importlib-resources-6.4.5
  Attempting uninstall: sqlalchemy
    Found existing installation: SQLAlchemy 2.0.36
    Uninstalling SQLAlchemy-2.0.36:
      Successfully uninstalled SQLAlchemy-2.0.36
  Attempting uninstall: typing-extensions
    Found existing installation: typing-extensions 4.12.2
    Uninstalling typing-extensions-4.12.2:
      Successfully uninstalled typing-extensions-4.12.2
ERROR: After October 2020 you may experience errors when installing or updating packages. This is because pip will change the way that it resolves dependency conflicts.
We recommend you use --use-feature=2020-resolver to test your packages with the new resolver before it becomes the default.

matplotlib 3.7.5 requires importlib-resources==3.2.0; python_version < "3.10", but you'll have importlib-resources 1.5.0 which is incompatible.
Successfully installed importlib-resources-1.5.0 sqlalchemy-1.4.54 typing-extensions-3.10.0.2
WARNING: You are using pip version 20.2.4; however, version 24.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.

airflow@a20fd9853b27: /opt/airflow$
```

Модули были установлены, так как много раз высказывали ошибки при запуске dag, анализировал логи, смотрел в каких местах ошибки и тем самым подгружал дополнительные библиотеки чтобы все заработало.

4. Статус запуска DAG:

The screenshot shows the Apache Airflow web interface in a Firefox browser. The address bar indicates the URL is `localhost:8080/home`. The interface includes a top navigation bar with links for DAGs, Security, Browse, Admin, and Docs. A status bar at the top right shows the time as 12:08 UTC. A notification banner at the top states: "Deleting DAG with id generate_data_dag. May take a couple minutes to fully disappear." Below this, the "DAGs" section is displayed. It features a filter input "Filter DAGs by tag" and a search input "Search DAGs". The DAGs are listed in a table with columns: DAG, Owner, Runs, Schedule, Last Run, Recent Tasks, Actions, and Links. Two DAGs are visible: "01_umbrella" and "employee_project_data_dag". The "employee_project_data_dag" has a last run time of "2024-10-19, 12:01:09". The interface also shows a pagination control at the bottom left and a version string at the bottom right: "Version: v2.0.0 Git Version: release-2.0.0-ab9f770b1cd8c690cbe4d0825896325aca0beeeca".

DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
01_umbrella	airflow	<div><div></div><div></div><div></div></div>	@daily		<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>
employee_project_data_dag	airflow	<div><div></div><div></div><div></div></div>	@daily	2024-10-19, 12:01:09	<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>	<div><div></div><div></div><div></div></div>

Два раза были неудачные запуски, где после установки дополнительных библиотек, корректировки кода все заработало.

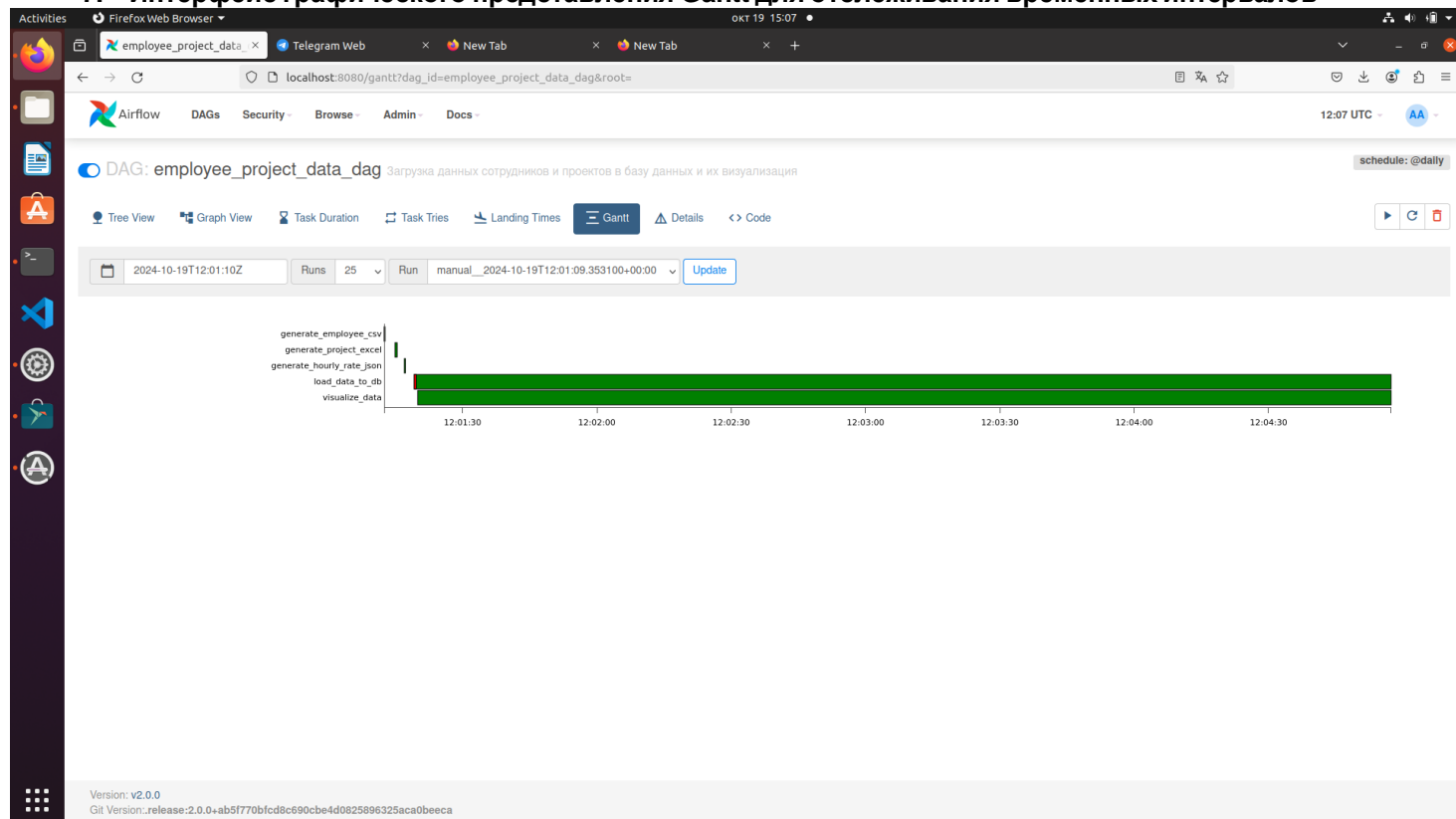
5. Интерфейс графического представления DAG :

The screenshot shows the Apache Airflow web interface in a Firefox browser. The URL is `localhost:8080/tree?dag_id=employee_project_data_dag`. The page title is "DAG: employee_project_data_dag" with a subtitle "Загрузка данных сотрудников и проектов в базу данных и их визуализация". The "Tree View" tab is selected. The interface shows a DAG diagram with five tasks: `generate_employee_csv`, `generate_project_excel`, `generate_hourly_rate_json`, `load_data_to_db`, and `visualize_data`. A legend at the bottom indicates task statuses: queued (grey), running (green), success (dark green), failed (red), up_for_retry (yellow), up_for_reschedule (light blue), upstream_failed (orange), skipped (pink), and scheduled (brown). The bottom status bar shows "Version: v2.0.0" and "Git Version: release:2.0.0+ab5f770bfc8c690cbe4d0825896325aca0beeca".

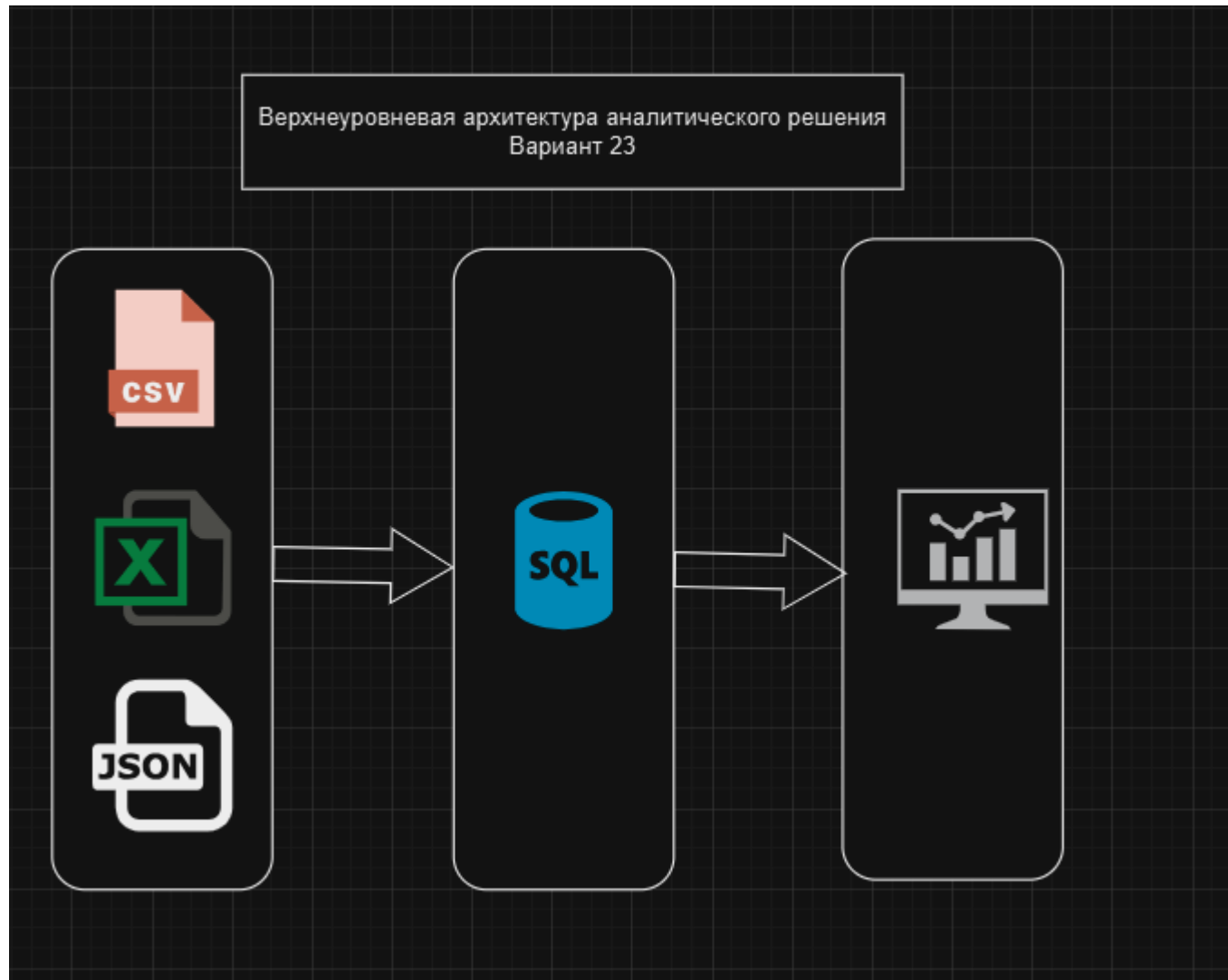
6. Интерфейс графического представления GraphView

The screenshot shows the Apache Airflow web interface in a Firefox browser, displaying the "Graph View" for the same DAG. The URL is `localhost:8080/graph?dag_id=employee_project_data_dag&root=`. The "Graph View" tab is selected. The DAG diagram is shown in a horizontal flow layout, with tasks `generate_employee_csv`, `generate_project_excel`, `generate_hourly_rate_json`, `load_data_to_db`, and `visualize_data` connected sequentially. A legend at the bottom indicates task statuses: queued (grey), running (green), success (dark green), failed (red), up_for_retry (yellow), up_for_reschedule (light blue), upstream_failed (orange), skipped (pink), and scheduled (brown). The bottom status bar shows "Version: v2.0.0" and "Git Version: release:2.0.0+ab5f770bfc8c690cbe4d0825896325aca0beeca".

7. Интерфейс графического представления Gantt для отслеживания временных интервалов



8.Спроектировать верхнеуровневую архитектуру аналитического решения задания 23



Вывод о проделанной работе и что получилось на выходе:

1. CSV файл с данными о сотрудниках**: Функция ``generate_employee_csv`` генерирует файл с данными о 100 сотрудниках, включая их имена, квалификацию и стаж. Данные варьируются с использованием ``random``, что делает каждый запуск уникальным.
Excel файл с данными о проектах**: Функция ``generate_project_excel`` создает Excel файл с данными о 20 проектах, где каждый проект включает 5 сотрудников и случайное количество часов работы.
JSON файл с данными о стоимости часа работы**: Функция ``generate_hourly_rate_json`` создает JSON файл, содержащий ставки за час работы для разных уровней квалификации.

2. Функция ``load_data_to_db`` загружает данные из всех трех форматов (CSV, Excel, JSON), объединяет их по сотрудникам и квалификациям, а затем записывает объединенные данные в базу данных SQLite. Это обеспечивает удобное хранение и доступ к данным.

3. Функция ``visualize_data`` извлекает данные из базы данных, группирует их по квалификации и подсчитывает общее количество часов работы для каждого уровня квалификации. Затем она создает барграф для визуализации этих данных, сохраняет его в виде изображения и закрывает фигуру.

4. Главный DAG ``employee_project_data_dag`` настроен для выполнения каждый день, начиная с 1 октября 2023 года. Он состоит из пяти задач, связанных между собой по порядку их выполнения.

5. Airflow используется для автоматизации процесса загрузки и обработки данных. Это особенно полезно для периодических заданий, где нужно генерировать, загружать и визуализировать данные на регулярной основе.

