

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

Вебинар 21 марта

Выполнил(а): st_105

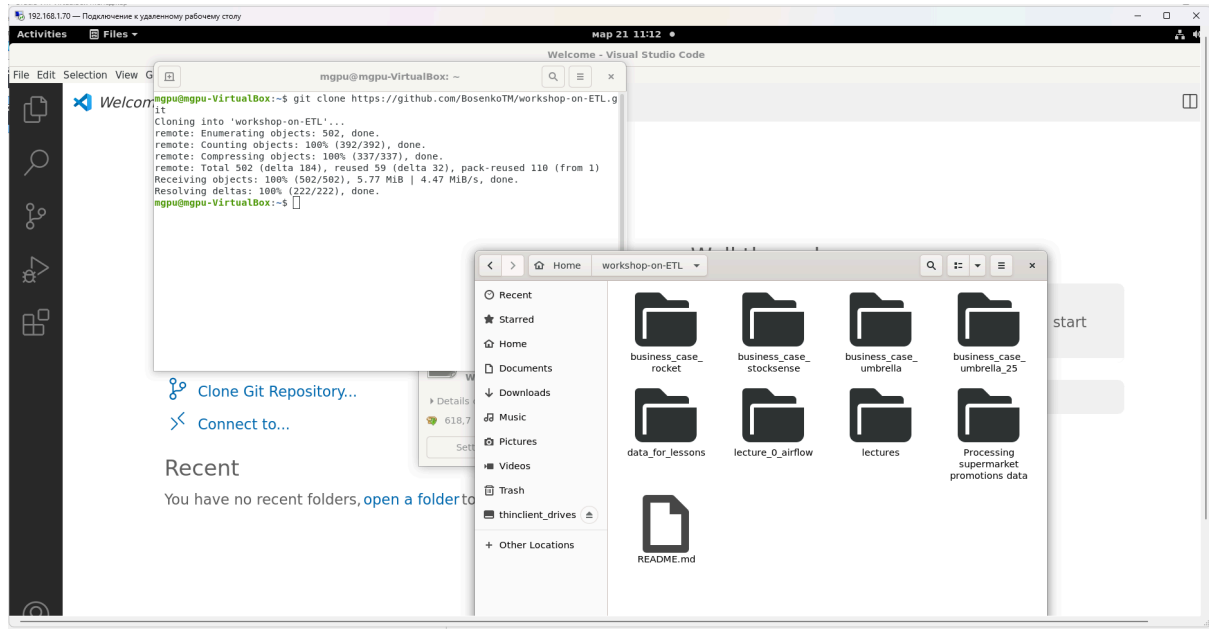
Москва

2025

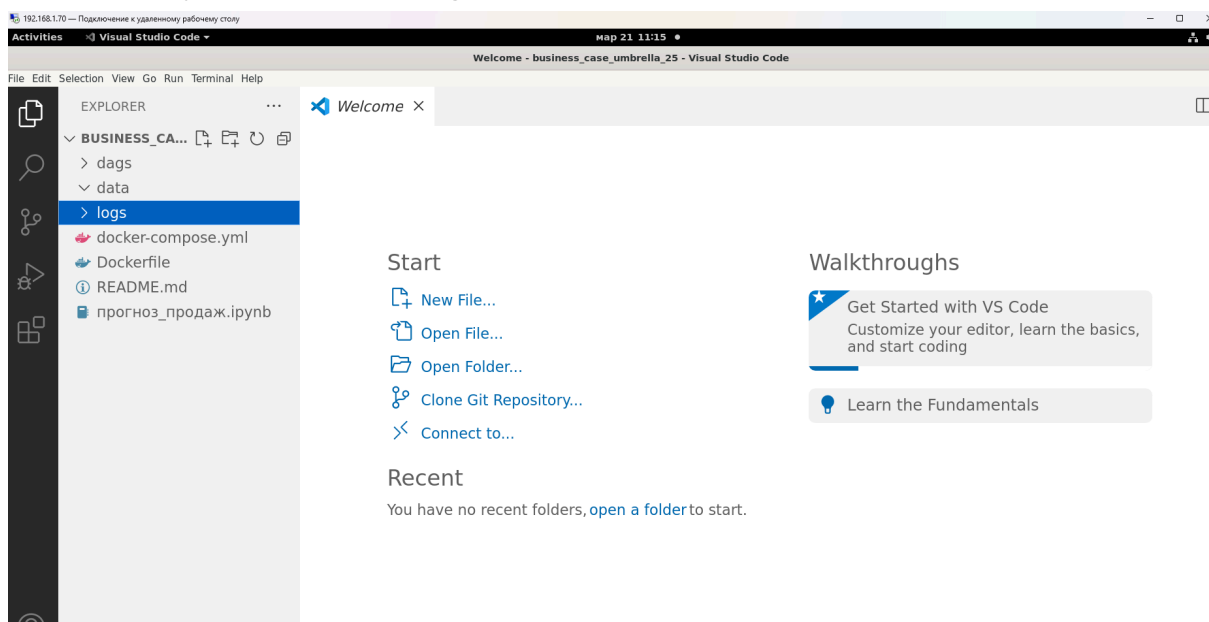
Бизнес кейс «Umbrella»

Общая часть:

скачивай репозиторий с помощью git clone



добавили пустые папки : logs и data



Проверяем первый даг,он не выводит результат,но показывает,что всё корректно

192.168.1.70 — Подключение к удаленному рабочему столу

Activities Firefox Web Browser Map 21 11:50 Dockerfile - business_case_umbrella_25 - Visual Studio Code

01_umbrella - Tree - AI x +

localhost:8080/tree?dag_id=01_umbrella

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back! Refresh Firefox...

Airflow DAGs Security Browse Admin Docs 08:50 UTC AU

Triggered 01_umbrella, it should start any moment now.

DAG: 01_umbrella Umbrella example with DummyOperators. schedule: @daily

Tree View Graph View Task Duration Task Tries Landing Times Gantt Details <> Code

2025-03-21T08:40:42Z Runs 25 Update

DummyOperator

queued running success failed up_for_retry up_for_reschedule upstream_failed skipped scheduled no_status

Mon 17 10:19 AM '25

[DAG]
fetch_weather_forecast
clean_forecast_data
join_datasets
train_ml_model
deploy_ml_model
fetch_sales_data
clean_sales_data
join_datasets

```
ic/pin_32.png HTTP/1.1" 304 (
untu; Linux x86_64; rv:136.0)
ng run <DagRun 01_umbrella @
0:00, externally triggered: 1
l: ttin
r with pid: 123
ARNING - Exception when impo
che-airflow-providers-microsc
.blob' (/home/airflow/.local
ARNING - Exception when impo
```

192.168.1.70 — Подключение к удаленному рабочему столу

Activities Firefox Web Browser Map 21 11:51 Dockerfile - business_case_umbrella_25 - Visual Studio Code

01_umbrella - Graph - x +

localhost:8080/graph?dag_id=01_umbrella

It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back! Refresh Firefox...

Airflow DAGs Security Browse Admin Docs 08:51 UTC AU

DAG: 01_umbrella Umbrella example with DummyOperators. success schedule: @daily

Tree View Graph View Task Duration Task Tries Landing Times Gantt Details <> Code

2025-03-21T08:50:43Z Runs 25 Run manual_2025-03-21T08:50:42.356484+00:00 Layout Left > Right Update Find Task...

DummyOperator

queued running success failed up_for_retry up_for_reschedule upstream_failed skipped scheduled no_status

Auto-refresh

fetch_weather_forecast → clean_forecast_data → join_datasets → train_ml_model → deploy_ml_model
fetch_sales_data → clean_sales_data → join_datasets

```
ic/dist/bootstrap-datetetimepic
ella" "Mozilla/5.0 (X11; Ubur
ic/dist/d3.min.js HTTP/1.1" 3
1; Ubuntu; Linux x86_64; rv::
ic/dist/d3-tip.js HTTP/1.1" 3
1; Ubuntu; Linux x86_64; rv::
ic/dist/taskInstances.5f1b6c
1_umbrella" "Mozilla/5.0 (X1
ic/dist/dagre-d3.min.js HTTP/
.0 (X11; Ubuntu; Linux x86_64
```

Регистрируемся на сайте weather api и копируем свой api

The image shows a screenshot of the WeatherAPI website. The top navigation bar includes links for Features, Pricing, API Explorer, Contact, and a My Account button. The main content area is titled 'Introduction' and describes the API's capabilities, such as providing real-time weather, forecasts, and historical data. A sidebar on the left lists various API features like Getting Started, Authentication, and Request URL. Below the introduction, there's a 'Welcome Back' section for a user, displaying their API key, plan status (Pro Plus), and trial end date. The dashboard also includes a 'Recent Activities' section and a 'Get Started' guide with links to documentation and tools.

weather api

Features Pricing API Explorer Contact My Account

Introduction

Getting Started
Authentication
Request URL
Request Param
Multilingual
Location Object
Weather Alerts
Air Quality **NEW**
Pollen **Coming Soon**
Weather Maps
Coming Soon
Bulk Request
API Error Codes

APIs

Realtime API

Introduction

WeatherAPI.com provides access to free weather and geo data via a JSON/XML restful API. It allows developers to create desktop, web and mobile applications using this data very easy.

We provide following data through our API:

- Real-time weather
- 14 day weather forecast
- Historical weather
- Marine Weather and Tide Data **NEW**
- Future Weather (Upto 300 days ahead) **NEW**
- Daily and hourly intervals
- 15 min interval **NEW** (Enterprise only)
- Astronomy
- Time zone
- Sports

Contact us

Dashboard

API

API Response Fields

Analytics

Accounts

Change Plan

Payment Method

Billing

Tools

API Explorer

Swagger Tool

Settings

Support

Change Password

Close Account

Log out

Welcome Back

API Key: f1d1b8c466274f69b5e85848252103 Copied LIVE TRIAL Ends on 04/Apr/2025

Pro Plus Plan

5,000,000 Calls per Month

0 Calls Made

04/Apr/25 Trial End Date

Note: If you are on a trial plan then after the trial plan ends your API key will be automatically moved to Free plan if you do not wish to upgrade to a paid plan.

☆ Get Started

- Learn how to form HTTP request to get weather from API Explorer or use our NEW Swagger Tool.
- Complete weather API documentation.
- Weather icons and weather lookup code list.
- Want to choose which weather field to return in the API response? Change it from API response fields.
- Looking to upgrade/downgrade your API plan? Visit our Upgrade/Downgrade plan section.

🕒 Regenerate API Key

Has your key been compromised? You can generate a new key.

Before you proceed?

- You will need to change your apps to use the new key.
- Your statistics will be reset.

Recent Activities

Account created 21-Mar-2025

Вносим изменения в файл
вставляем свой api

```

}
# 1. Получение прогноза погоды
def fetch_weather_forecast():
    api_key = "f1d1b8c466274f69b5e85848252103" # замените на ваш API ключ
    url = f"http://api.weatherapi.com/v1/forecast.json?key={api_key}&q=Londo"
    response = requests.get(url)
    data = response.json()
    forecast_data = [(day['date'], day['day']['avgtemp_c']) for day in data['forecast']]
    df = pd.DataFrame(forecast_data, columns=['date', 'temperature'])
    data_dir = '/opt/airflow/data'
    os.makedirs(data_dir, exist_ok=True)
    df.to_csv(os.path.join(data_dir, 'weather_forecast.csv'), index=False)

```

Данные продаж устанавливаем за 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'],
        'sales': [100, 150, 200, 75, 130, 275, 108]
    }
    df = pd.DataFrame(sales_data)

```

Вариант 15

Получить прогноз в Бангкоке на 7 дней	Сравнить температуры первого и последнего дня	Вывести результат на экран
---------------------------------------	---	----------------------------

Для получения данных из Бангока редактируем Dag

```

23
24 Получение прогноза погоды для Бангкока
25 def fetch_weather_forecast():
26     api_key = "f1d1b8c466274f69b5e85848252103" # замените на ваш API ключ
27     url = f"http://api.weatherapi.com/v1/forecast.json?key={api_key}&q=Bangkok&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']]
31     df = pd.DataFrame(forecast_data, columns=['date', 'temperature'])
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

```

Чтобы в логах выводились температуры первого и второго дня добавила в лог:

```
Сравнение температур первого и последнего дня
compare_temperatures(**kwargs):
data_dir = '/opt/airflow/data'
df = pd.read_csv(os.path.join(data_dir, 'weather_forecast.csv')

# Температура первого дня
first_day_temp = df.iloc[0]['temperature']

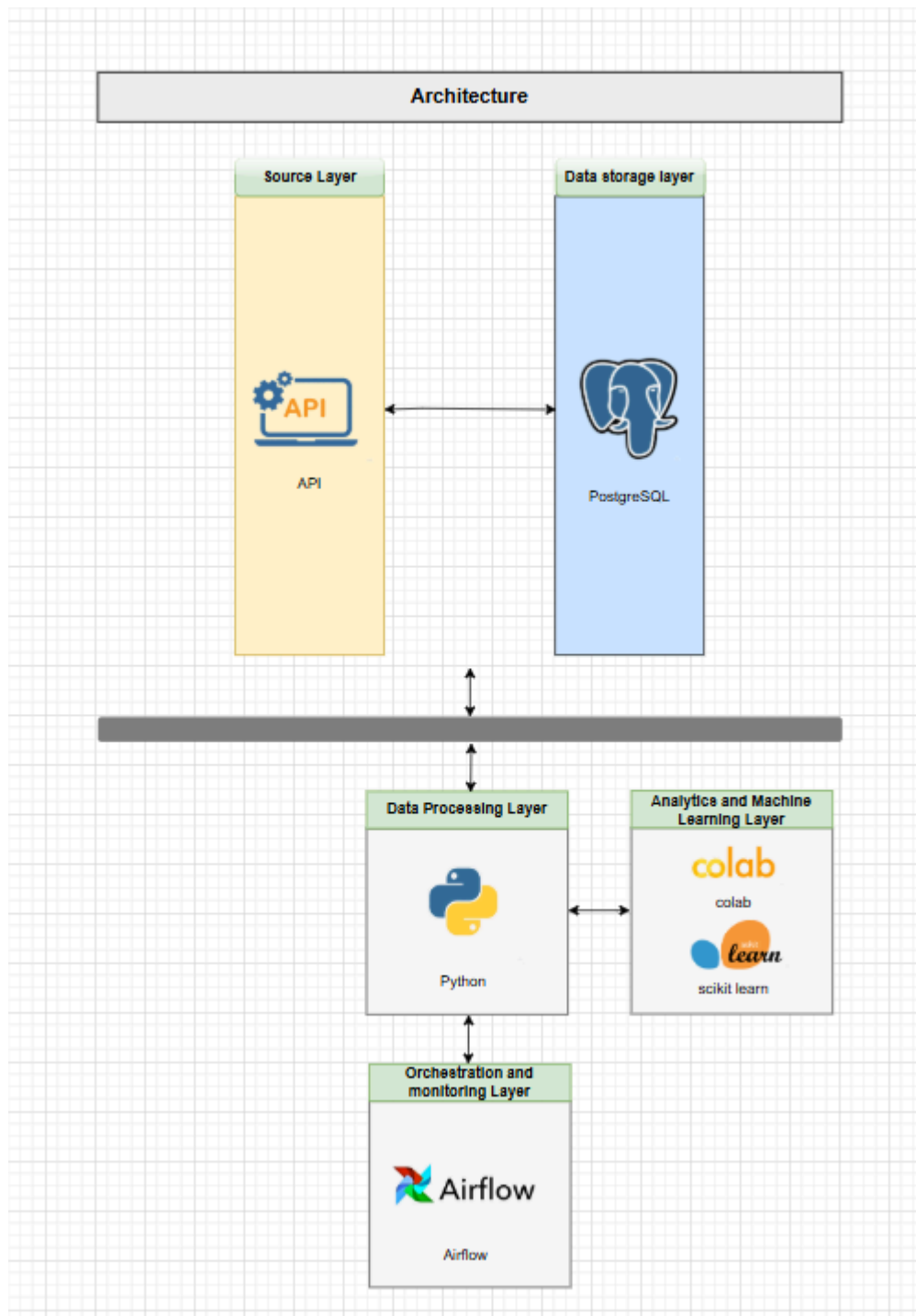
# Температура последнего дня
last_day_temp = df.iloc[-1]['temperature']

# Вывод результата
print(f"Температура первого дня: {first_day_temp}°C")
print(f"Температура последнего дня: {last_day_temp}°C")

if first_day_temp > last_day_temp:
```

Также сравнение температур было выполнено в google colab.

Составлена архитектура:



Выводы по работе:

В ходе работы было выполнено подключение через api и с помощью этого получены данные для обучения модели. Составлена верхнеуровневая архитектура, а также выполнена визуализация результатов в google.colab <https://colab.research.google.com/drive/16AuELI4CeaWvSkRnaf6nQxzeuxiqT-UQ?usp=sharing>