



## Разработка виджета для отображения текущих погодных данных TrafficIQ

Горбушин Алексей Алексеевич  
Малащенко Мария Степановна

# Определение области проекта

Транспорт и дорожное движение – интеграция виджета, отображающего текущую погоду, для анализа и отображения дорожной ситуации в реальном времени.

## Обстановка на дороге

(используются открытые данные)

Небольшой дождь



+2°



6 м/с

# Сбор данных

\*Пример структуры JSON

Источник данных - <https://api.openweathermap.org/data/2.5/weather> -  
бесплатный сайт погоды OpenWeatherMap, открытый API.

```
{
  "coord": {
    "lon": 37.9,
    "lat": 59.1
  },
  "weather": [
    {
      "id": 804,
      "main": "Clouds",
      "description": "пасмурно",
      "icon": "04d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 3,
    "feels_like": -0.32,
    "temp_min": 3,
    "temp_max": 3,
    "pressure": 1017,
    "humidity": 100,
    "sea_level": 1017,
    "grnd_level": 1002
  },
  "wind": {
    "speed": 3.57,
    "deg": 244,
    "gust": 7.77
  },
  "clouds": {
    "all": 100
  },
  "dt": 1765969621,
  "sys": {
    "type": 1,
    "id": 8931,
    "country": "RU",
    "sunrise": 1765952359,
    "sunset": 1765974585
  },
  "timezone": 10800,
  "id": 569223,
  "name": "Неревноец",
  "cod": 200
}
```

Получение текущей погоды

```
{
  "cod": "200",
  "message": 0,
  "cnt": 40,
  "list": [
    {
      "dt": 1765972800,
      "main": {
        "temp": 3,
        "feels_like": -0.31,
        "temp_min": 1.23,
        "temp_max": 3,
        "pressure": 1017,
        "sea_level": 1017,
        "grnd_level": 1003,
        "humidity": 99,
        "temp_kf": 1.77
      },
      "weather": [
        {
          "id": 804,
          "main": "Clouds",
          "description": "пасмурно",
          "icon": "04d"
        }
      ],
      "clouds": {
        "all": 100
      },
      "wind": {
        "speed": 3.56,
        "deg": 245,
        "gust": 7.92
      },
      "pop": 0,
      "sys": {
        "pod": "d"
      },
      "dt_txt": "2025-12-17 12:00:00"
    }
  ],
  //продолжение по каждому часу
},
"city": {
  "id": 569223,
  "name": "Неревноец",
  "coord": {
    "lat": 59.1,
    "lon": 37.9
  },
  "country": "RU",
  "population": 311850,
  "timezone": 10800,
  "sunrise": 1765952359,
  "sunset": 1765974585
}
```

Получение прогнозных данных

# Создание базы данных

1. Создание БД (выделение сервера, установка PostgreSQL)

```
CREATE DATABASE "meteo" WITH OWNER = "postgres";  
CREATE SCHEMA "meteo" AUTHORIZATION "postgres";
```

2. Параметры БД

|          |                 |
|----------|-----------------|
| host     | "192.168.*.*"   |
| port     | 5432            |
| database | "meteo"         |
| user     | "user_name"     |
| password | "user_password" |

# Парсинг файлов-источников, разбор структуры и загрузка данных в БД



Подключение к сайту openweathermap.org осуществляется через API сайта

1. Метод <https://api.openweathermap.org/data/2.5/weather?> (получаемый json для текущей погоды)
2. Метод <https://api.openweathermap.org/data/2.5/forecast?> (получаемый json Прогноз)
3. Параметры для методов:

|                             |   |
|-----------------------------|---|
| lat = 59.9                  | # долгота в десятичном формате  |
| lon = 30.3                  | # широта в десятичном формате   |
| exclude = 'minutely,alerts' | # исключения без пробелов через запятую<br>current,minutely,hourly,daily,alerts |
| lang = 'ru                  | # язык для комментариев   |
| units = 'metric'            | # 'standard' - Кельвин, 'metric' - Цельсий, 'imperial' - Фаренгейт              |

Обращение к api-методам осуществляется через Python-скрипт. В самом скрипте выполняется передача данных, полученных с сайта, в БД с помощью библиотеки psycopg2

## Файл - get\_api.py

Из скрипта вызываются хранимые процедуры, которые располагаются на БД и выполняют запись данных.

Здесь производится проверка данных на корректность с точки зрения целостности БД (например, запись не будет произведена, если придет категория погоды, для которой нет соответствия легенды, направления ветра), а также с точки зрения физического соответствия.

Например, превышение температуры окружающего воздуха значений 50°C и выше

## Хранение данных

Выбрана реляционная БД PostgreSQL 12.22

В БД созданы 2 таблицы:

1. Таблица CURRENT\_WEATHER, содержащая строки, получаемые из JSON-файла текущей погоды
2. Таблица FORECAST\_HOURLY, содержащая строки, получаемые из JSON-файла прогноза погоды

За безопасное хранение данных отвечает СУБД и обеспечивается расположением на виртуальном сервере с регулярным резервным копированием.

# Создание таблиц БД

| Name                | Type      | Length | Dec | Not NULL | Key | Description   |
|---------------------|-----------|--------|-----|----------|-----|---|
| id                  | int4      | 32     | 0   | Да       | Да  |   |
| date                | timestamp | 6      | 0   | Да       |     | дата измерений  |
| weather_id          | int2      | 16     | 0   |          |     | Идентификатор погодных условий  |
| weather_main        | varchar   | 50     | 0   |          |     | Группа погодных параметров (Дождь, Снег, Экстрим и др.)   |
| weather_description | varchar   | 50     | 0   |          |     | Погодные условия в группе на своем языке  |
| weather_icon        | varchar   | 10     | 0   |          |     | Идентификатор значка погоды   |
| current_temp        | numeric   | 6      | 2   |          |     | Температура   |
| current_feels_like  | numeric   | 6      | 2   |          |     | как ощущается   |
| current_pressure    | numeric   | 6      | 2   |          |     | Атмосферное давление, гПа   |
| current_humidity    | numeric   | 6      | 2   |          |     | Влажность, %  |
| current_visibility  | int2      | 16     | 0   |          |     | видимость   |
| current_wind_speed  | numeric   | 6      | 2   |          |     | Скорость ветра, метр / сек  |
| current_wind_deg    | numeric   | 6      | 2   |          |     | Направление ветра, градусы  |
| current_clouds      | numeric   | 6      | 2   |          |     | Облачность, %   |
| current_rain_1h     | numeric   | 6      | 2   |          |     | Объем дождя за последний час, мм  |
| current_snow_1h     | numeric   | 6      | 2   |          |     | Объем снега за 1 час, мм  |
| current_sunrise     | timestamp | 6      | 0   |          |     | Время восхода, unix, UTC  |
| current_sunset      | timestamp | 6      | 0   |          |     | Время заката, unix, UTC   |
| current_uvi         | numeric   | 6      | 2   |          |     | Текущий УФ-индекс   |
| current_dew_point   | numeric   | 6      | 2   |          |     | Атмосферная температура (меняется в зависимости от давления и влажности), ниже которой капли воды начинают конденсироваться и может образовываться роса |
| current_wind_gust   | numeric   | 6      | 2   |          |     | Порыв ветра. Единицы - по умолчанию: метр / сек, метрическая система: метр / сек, британская: мили / час.   |

CURRENT\_WEATHER (Текущая погода)

Файл - Create\_table\_current\_weather.sql

| Name                | Type      | Length | Dec | Not NULL | Key | Description   |
|---------------------|-----------|--------|-----|----------|-----|---|
| id                  | int4      | 32     | 0   | Да       | Да  | Идентификатор   |
| date                | timestamp | 6      | 0   | Да       |     | дата измерений  |
| hourly_temp         | numeric   | 6      | 2   |          |     | Температура. Units – default: kelvin, metric: Celsius, imperial: Fahrenheit.  |
| hourly_feels_like   | numeric   | 6      | 2   |          |     | Температура. Человеческое восприятие погоды, ощущается как. Units – default: kelvin, metric: Celsius, imperial: Fahrenheit.   |
| hourly_pressure     | numeric   | 6      | 2   |          |     | Атмосферное давление на уровне моря, hPa  |
| hourly_humidity     | numeric   | 6      | 2   |          |     | Влажность, %  |
| hourly_dew_point    | numeric   | 6      | 2   |          |     | Атмосферная температура (изменяющаяся в зависимости от давления и влажности), ниже которой капли воды начинают конденсироваться и может образовываться роса. Units – default: kelvin, metric: Celsius, imperial: Fahrenheit |
| hourly_uvi          | numeric   | 6      | 2   |          |     | УФ-индекс   |
| hourly_clouds       | numeric   | 6      | 2   |          |     | Облачность, %   |
| hourly_visibility   | int2      | 16     | 0   |          |     | Средняя видимость, metres   |
| hourly_wind_speed   | numeric   | 6      | 2   |          |     | Скорость ветра. Units – default: metre/sec, metric: metre/sec, imperial: miles/hour   |
| hourly_wind_gust    | numeric   | 6      | 2   |          |     | Порывы ветра. Units – default: metre/sec, metric: metre/sec, imperial: miles/hour.  |
| hourly_wind_deg     | numeric   | 6      | 2   |          |     | Направление ветра, градусы  |
| hourly_pop          | numeric   | 6      | 2   |          |     | Вероятность осадков   |
| hourly_rain_1h      | numeric   | 6      | 2   |          |     | Объем осадков - дождя за прошлый час, мм  |
| hourly_snow_1h      | numeric   | 6      | 2   |          |     | Объем осадков - снега за прошлый час, мм  |
| weather_id          | int2      | 16     | 0   |          |     | Идентификатор погодных условий  |
| weather_main        | varchar   | 50     | 0   |          |     | Группа погодных условий (Rain, Snow, Extreme etc.)  |
| weather_description | varchar   | 50     | 0   |          |     | Погодные условия внутри группы (Дождь - Легкий дождь)   |
| weather_icon        | varchar   | 10     | 0   |          |     | Идентификатор иконки погоды   |

FORECAST\_HOURLY (почасовой прогноз)

Файл - Create\_table\_forecast\_hourly.sql

# Предобработка данных



Алгоритм предполагает, что почасовой прогноз, записанный в предыдущий раз, удаляется (очищается из таблицы), затем заменяется на новый, пришедший на текущее время.

**Файл - meteo\_forecast\_hourly.sql**

Текущая погода, реальное состояние дописывается в таблицу актуальной погоды, там будет храниться фактическая погода за время, определенное потребностями задачи.

**Файл - meteo\_current\_weather.sql**

Погодные данные приходят с кодами для погодного состояния и направления ветра. Поэтому для корректного пользовательского отображения погоды нам необходимы: картинки погоды или их интерпретации (иконки) и расшифровка направления ветра. Для них создаются две отдельные таблицы, которые наполняются отдельно с сайта единого разово.



# Создание таблиц-справочников БД

| Name        | Type    | Length | Dec | Not NULL | Key | Description    |
|-------------|---------|--------|-----|----------|-----|----------------|
| id          | int4    | 32     | 0   | Да       | Да  | Идентификатор  |
| group       | int4    | 32     | 0   |          |     | Группа         |
| main        | varchar | 0      | 0   |          |     | Наименование   |
| description | varchar | 0      | 0   |          |     | Описание       |
| icon_day    | varchar | 0      | 0   |          |     | Иконка дневная |
| icon_night  | varchar | 0      | 0   |          |     | Иконка ночная  |

WEATHER\_LEGEND (легенда, расшифровка погоды для отображения)

Пример данных:

|     |   |        |                                     |     |     |
|-----|---|--------|-------------------------------------|-----|-----|
| 230 | 2 | гроза  | гроза с мелким дождем               | 11d | 11n |
| 231 | 2 | гроза  | гроза с моросью                     | 11d | 11n |
| 232 | 2 | гроза  | гроза с сильным морозящим дождем    | 11d | 11n |
| 300 | 3 | морось | небольшая морось                    | 09d | 09n |
| 301 | 3 | морось | морось                              | 09d | 09n |
| 302 | 3 | морось | сильный дождь                       | 09d | 09n |
| 310 | 3 | морось | интенсивность света морозящий дождь | 09d | 09n |
| 311 | 3 | морось | морозящий дождь                     | 09d | 09n |
| 312 | 3 | морось | сильный морозящий дождь             | 09d | 09n |
| 313 | 3 | морось | ливень, дождь и изморось            | 09d | 09n |
| 314 | 3 | морось | сильный ливень, дождь и изморось    | 09d | 09n |
| 321 | 3 | морось | изморось                            | 09d | 09n |

Файл - Create\_table\_weather\_legend.sql

| Name          | Type    | Length | Dec | Not NULL | Key | Description                           |
|---------------|---------|--------|-----|----------|-----|---------------------------------------|
| id            | int4    | 32     | 0   | Да       | Да  |                                       |
| wind_deg_from | int4    | 32     | 0   |          |     | Градус-начало сектора                 |
| wind_deg_to   | int4    | 32     | 0   |          |     | Градус-окончание сектора              |
| name          | varchar | 255    | 0   |          |     | Наименование направления ветра        |
| full_name     | varchar | 255    | 0   |          |     | Полное наименование направления ветра |
| order_num     | int4    | 32     | 0   |          |     | № п/п                                 |

WIND\_DEG (расшифровка направления ветров)

Пример данных:

|   |     |     |    |               |   |
|---|-----|-----|----|---------------|---|
| 1 | 0   | 22  | C  | Север         | 1 |
| 2 | 23  | 67  | CB | Северо-Восток | 2 |
| 3 | 68  | 112 | B  | Восток        | 3 |
| 4 | 113 | 157 | ЮВ | Юго-Восток    | 4 |
| 5 | 158 | 202 | Ю  | Юг            | 5 |
| 6 | 203 | 247 | ЮЗ | Юго-Запад     | 6 |
| 7 | 248 | 292 | З  | Запад         | 7 |
| 8 | 293 | 337 | СЗ | Северо-Запад  | 8 |
| 9 | 338 | 360 | С  | Север         | 1 |

Файл - Create\_table\_wind\_deg.sql

# Разметка данных



На данный момент объем данных небольшой. При необходимости работы с данными за большой период (несколько лет) можно разделить таблицу на секции (партиции) с разбиением по ключу ID+ год, месяц.

Для пользователя использование таблиц будет прозрачным.

# Интеграция данных



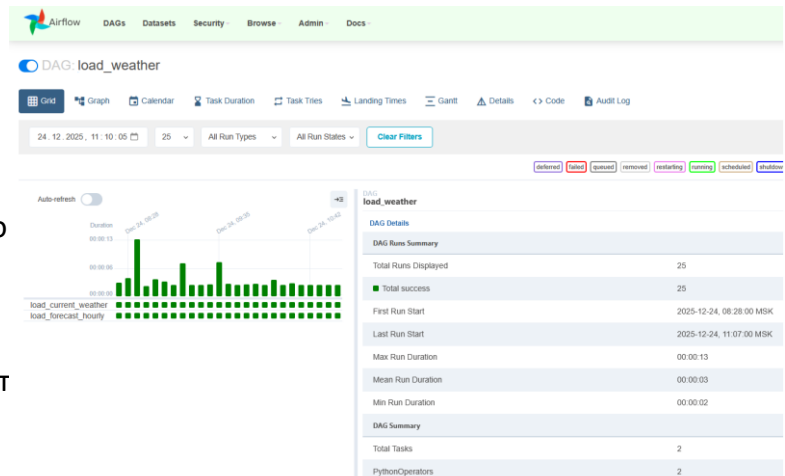
1. Загрузка данных с API происходит автоматическим путем каждые 5 минут.

Автоматизация процесса состоит в том, чтобы переложить запуск написанного нами Python-скрипта (get\_api.py) на:

- .bat/.py-файл, который будет запускаться каждое определенное число часов/минут/секунд и работать на сервере под наблюдением и администрированием;
- любой планировщик, который способен запускать скрипты, установленный на сервере, где есть Python, выход в интернет и открыт доступ к внутренней сети и нашей БД (как правило, она устанавливается на другом сервере);
- скрипт запускает написанные на БД хранимые процедуры, для этого мы создаем специального пользователя с доступом.

В нашем случае мы используем Apache Airflow Version: v2.5.0.

Для того, чтобы Python-файл адаптировать для запуска из AirFlow (он работает с DAG-скриптами), в заголовке снимаем комментарии с подключения библиотеки Airflow и комментарии вызова. И комментируем вызов функций синтаксиса Python.



# Интеграция данных

2. Встраивание виджета на сайт (в другую систему или в качестве погодного модуля) происходит на этапе разработки системы.

## Обстановка на дороге

(используются открытые данные)

Небольшой дождь



+2°



6 м/с

```
def get_weather_data() -> WeatherData:
    with psycopg2.connect(
        user=DB_USER,
        password=DB_PASSWORD,
        dbname=DB_NAME,
        host=DB_HOST,
        port=DB_PORT,
    ) as conn:
        with conn.cursor() as cursor:
            cursor.execute(
                """
                SELECT './assets/'
                || REPLACE(b.weather_description, ' ','-') || '.png'
                || CASE WHEN date_part('MONTH', b.date) IN (1,2,3,4,10,11,12) THEN 'зима_'
                ELSE 'лето_' END
                || CASE WHEN date_part('HOUR', b.date) IN (0,1,2,3,22,23,24) THEN 'ночь_'
                WHEN date_part('HOUR', b.date) IN (4,5,6,7,8,9) THEN 'утро_'
                WHEN date_part('HOUR', b.date) IN (10,11,12,13,14,15) THEN 'день_'
                ELSE 'вечер_' END
                || '.png' AS icon_file,* FROM "meteo"."get_current_weather"() b
                """
            )
            result = cursor.fetchone()

            return WeatherData(*result)
```

# Очистка данных



Данные сайта погоды OpenWeatherMap достаточно достоверны для того, чтобы мы их размещали напрямую в БД. На сайте работают собственные механизмы проверки данных, которые позволяют пользоваться ими без дополнительного «приземления данных». Очистка, проверка данных на недостоверность не требуется.

Обычный порядок работы с данными таков:

- Получение и «приземление» данных, размещение их в таблице/таблицах, соответствующих структурам получаемых данных, с отслеживанием даты загрузки, возможно, порции, и прочим.
- Парсинг и очистка данных, возможно, выделение только необходимых данных, с проставлением даты обработки, признака использования.
- Перенос обработанных данных/части данных в целевые таблицы, годные для использования этих данных. Возможно, выделение справочной информации, сопоставление изменений с существующими данными и обновление их по оговоренным алгоритмам.
- Работа с целевыми таблицами – запросы приложений.

# Обеспечение выполнения бизнес-правил



А также необходимо отметить, что очистка данных может производиться не только на уровне физического соответствия структуре БД, но и бизнес-правилам, например:

- В поле целого числа (например, мы могли скачивать скорость ветра – 1м/с, 2 м/с и так далее), определив это поле как целое число. Гипотетически (например, после обновления сайта, улучшения оборудования, изменения региона) мы можем получить значение для этого поля типа decimal, например, 2.5 м/с. Это ошибка **несоответствия структуре данных БД**. Ошибку при загрузке мы получим от БД.
- Для прогноза мы берем данные только на будущий час. Неважно, что сайт отдает на предыдущий день, но нам нужны только на будущий час. Эти условия мы должны реализовать самостоятельно, отсеяв строки за другие часы. Ошибки здесь мы не получим, но **бизнес-правилу эти данные не соответствуют**, их не должно быть в целевой таблице.
- Мы получили температуру воздуха +50, хотя сейчас декабрь, и город СПб. Это бизнес-правило относится к тем, которые играют роль исключительно в определенных условиях. Эта температура окружающего воздуха вполне допустима в других условиях, например, в пустыне, на Экваторе, в сауне и т.д., но здесь мы отнесем такое значение к **ошибке по бизнес-правилу**.

В нашем примере мы используем дополнение «очистки данных» - проверку по бизнес-правилам на превышение температуры. Включим ее на уровне БД – не будем записывать «ошибочные» данные, допишем условия.

# Обеспечение выполнения бизнес-правил



```
-- дописываем к скрипту "meteo"."add_current_weather"  
WHERE  
-- температура воздуха, C  
| ((t.val::json -> 'main')::json ->> 'temp')::DECIMAL(6,2) < 50 AND  
-- скорость ветра, м/с  
| (t.val::json -> 'wind' ->> 'speed')::DECIMAL(6,2) < 20
```

Обеспечение БП для текущей погоды

```
-- дописываем к скрипту "meteo"."add_forecast_hourly"  
WHERE  
-- температура воздуха, C  
| (jd.val -> 'main' ->> 'temp')::DECIMAL(6,2) < 50 AND  
-- скорость ветра, м/с  
| (jd.val -> 'wind' ->> 'speed')::DECIMAL(6,2) < 20
```

Обеспечение БП для прогноза погоды

## Анализ данных и визуализация

Визуализация выполняется с помощью Apache Superset 2.0.1

Отображение текущей погоды:

```
SELECT * FROM "meteo"."get_current_weather"()
```

Содержимое функции:

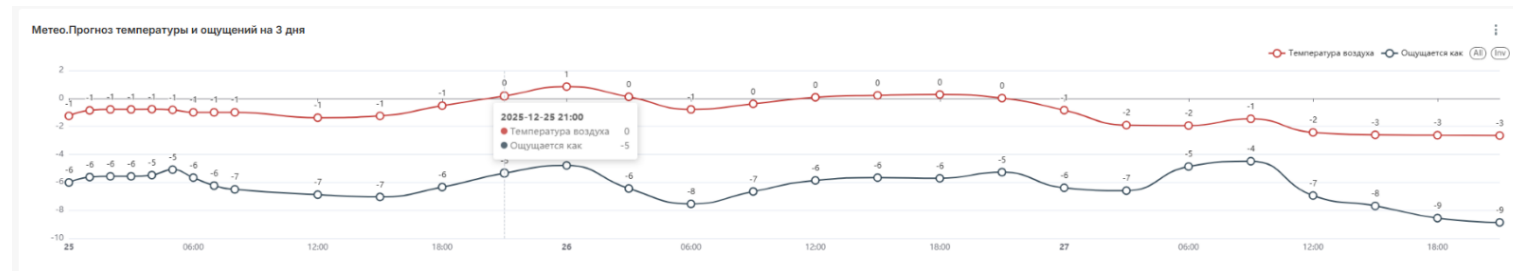
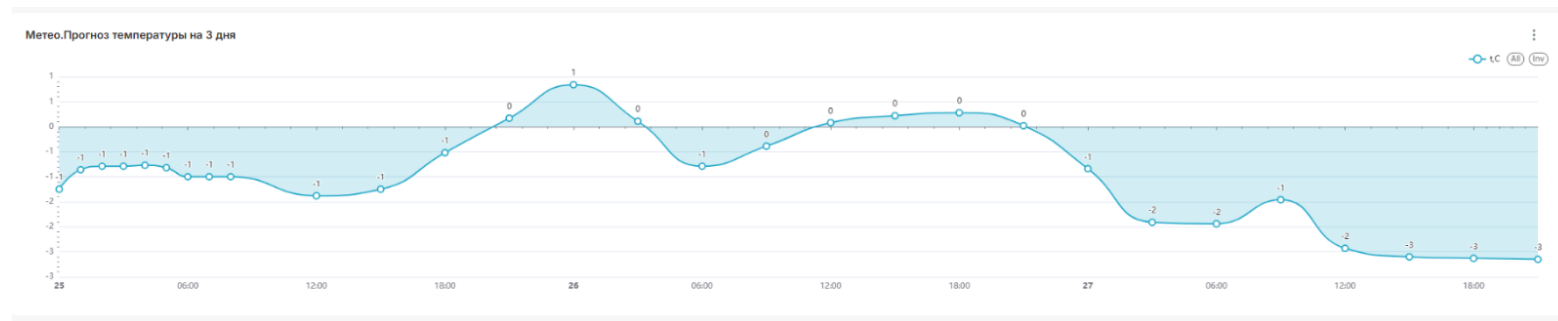
Файл - `get_current_weather.sql`



# Дашборды



## Прогноз погоды на ближайшие 3 дня



```
SELECT * FROM "meteo"."get_hourly_weather"()
```

# Дашборды

## Направление ветра

Метео. Прогноз. Скорость ветра

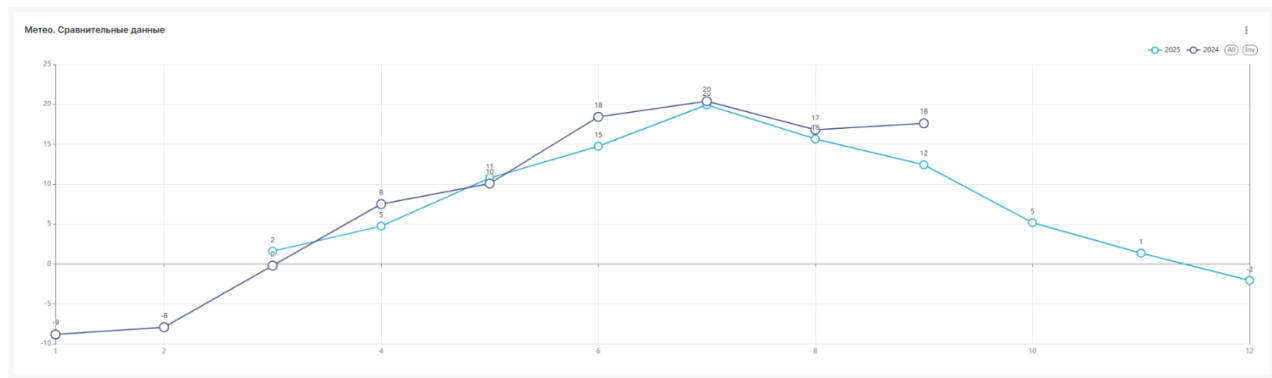


```
SELECT hourly_wind_desc AS hourly_wind_desc,  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = 'C') AS "C",  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = 'C3') AS "C3",  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = 'CB') AS "CB",  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = 'B') AS "B",  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = 'ЮВ') AS "ЮВ",  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = 'Ю') AS "Ю",  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = 'Ю3') AS "Ю3",  
MIN(hourly_wind_speed) FILTER (WHERE hourly_wind_desc = '3') AS "3"  
FROM  
(SELECT date + make_interval(hours => hour) AS dd, * FROM "meteo"."get_hourly_weather"()  
 ) AS virtual_table GROUP BY hourly_wind_desc ORDER BY "C" DESC
```

# Дашборды



## Сравнение средних температур за месяц



```
SELECT EXTRACT('MONTH' FROM "date")::INTEGER AS month, AVG(current_temp) AS "2025"
FROM (SELECT
    cw.id,
    cw.date::TIMESTAMP,
    cw.weather_id,
    cw.current_temp::DECIMAL(6,2),
    cw.current_feels_like,
    cw.current_wind_speed
    FROM meteo.current_weather      cw
    WHERE
        cw.date >= '2024-01-01 00:00:00' AND
        cw.date <= '2026-01-01 00:00:00'
) AS virtual_table
WHERE EXTRACT('YEAR' FROM "date")::INTEGER = '2025' GROUP BY EXTRACT('MONTH' FROM "date")::INTEGER ORDER BY "2025" DESC
LIMIT 1000;
```

# Команда



Алексей Горбушин

1. Настройка среды разработки и сервера
2. Работа с API OpenWeatherMap
3. Создание и поддержание схемы базы данных для долгосрочного хранения и быстрого доступа к данным.
4. Анализ качества данных
5. Мониторинг поступающих данных на предмет ошибок и аномалий
6. Настройка AirFlow



Мария Малащенко

1. Сбор и преобразование данных
2. Разработка процессов очистки и преобразования сырых данных в удобный для анализа формат.
3. Реализация примера динамического погодного виджета, интегрированного с базой данных.
4. Создание отчётов и дашбордов в Apache Superset для визуализации.

Спасибо  
за внимание!

it's **MO**re than a  
**UNIVERSITY**