МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики
Кафедра технической кибернетики

Отчет по лабораторной работе №1

Дисциплина: «Инженерия данных»

Тема: **«Введение в ETL пайплайны | Знакомство с Prefect и ClickHouse»**

Выполнил: Казаркина Д.Д.

Группа: 6233-010402D

Самара 2025

ЗАДАНИЕ

1. В качестве источника данных предлагается использовать Free Weather API.
2. (Extract) Получить прогноз на завтра по переменным: температура, осадки, скорость и направление ветра для городов Самара и Москва. Сырые ответы API сохранить в объектном хранилище.
3. (Transform)
   - Извлечь почасовые значения и нормализовать для таблицы weather_hourly.
   - Посчитать дневную статистику (min, max, avg температуру и количество осадков) и подготовить для сохранения в таблице weather_daily.
4. (Load) Загрузить преобразованные данные в соответствующие таблицы ClickHouse.
5. Автоматически отправить уведомления в Telegram с кратким прогнозом на завтра и предупреждать о сильном ветре/осадках.
6. (Опционально) Реализовать обработку различных ошибок.

ЛАБОРАТОРНАЯ РАБОТА

1. Техническое описание пайплайна

Архитектура. Система реализует ETL-пайплайн с использованием Prefect для оркестрации задач. Сырые данные сохраняются в S3-совместимое хранилище MinIO, преобразуются и загружаются в аналитическую СУБД ClickHouse. Уведомления отправляются через Telegram Bot API. Все компоненты контейнеризованы через Docker Compose, что обеспечивает воспроизводимость окружения.

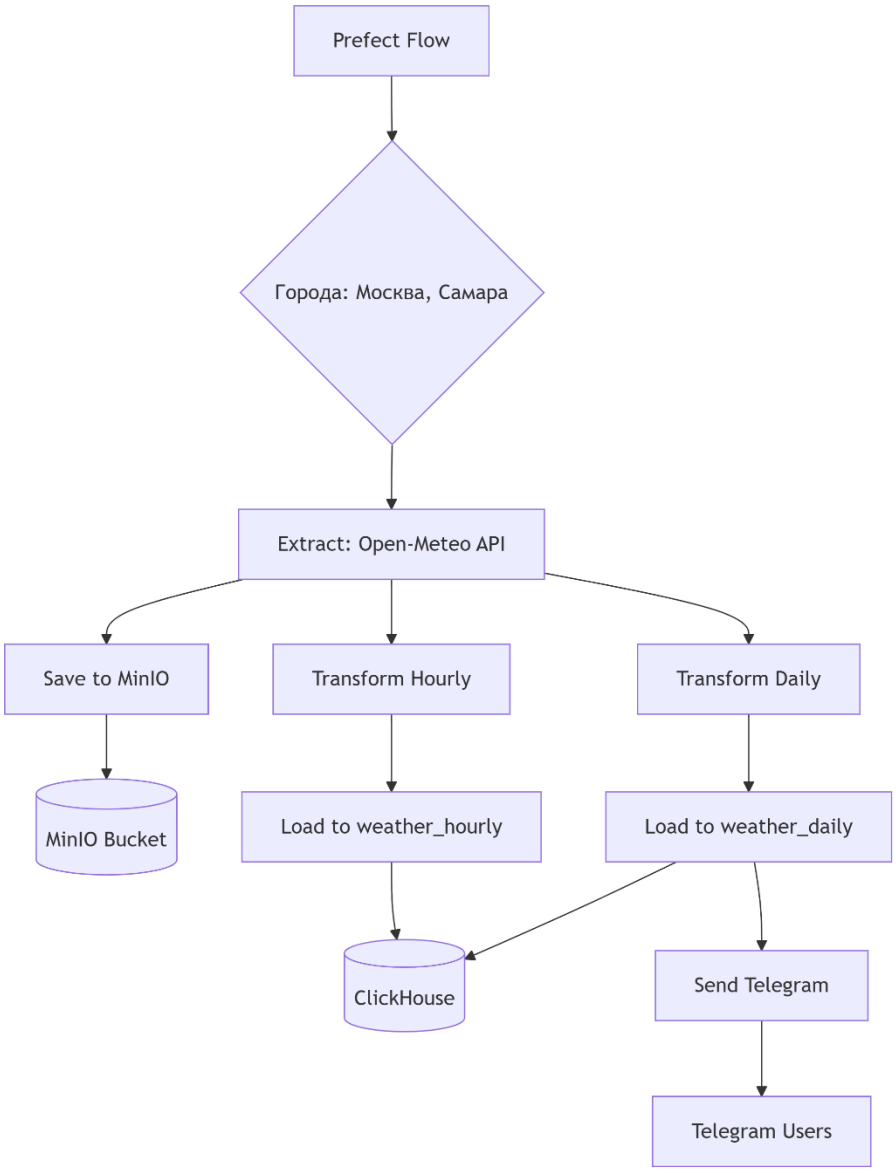


Рисунок 1 – Схема пайплайна

Источник данных. Использовано бесплатное API Open-Meteo. Для каждого города отправляется GET-запрос на эндпоинт https://api.open-meteo.com/v1/forecast с параметрами: координаты (широта, долгота), почасовые данные температуры, осадков и ветра на 2 дня, автоматический часовой пояс. API возвращает JSON с прогнозом на 48 часов.

Extract. Для каждого города модуль отправляет GET-запрос к API Open-Meteo, запрашивая почасовой прогноз на два дня. Из 48-часового ответа выделяются данные на завтра (часы 24-48), к ним добавляются метаполя city и fetched_at, а затем сырой JSON сохраняется в MinIO по пути {город}/{дата}.json.

Transform. Сырой JSON преобразуется в список плоских почасовых записей с полями город, дата, час, температура, осадки и ветер. На основе этих записей вычисляются дневные агрегаты: минимальная, максимальная и средняя температура, сумма осадков и максимальная скорость ветра.

Load. Почасовые записи bulk-вставкой загружаются в таблицу ClickHouse weather_hourly, а агрегированные дневные статистики – в таблицу weather_daily. Для загрузки используется драйвер clickhouse-driver, обеспечивающий эффективную вставку данных.

Качество данных. Реализованы проверки HTTP-статуса ответа API, обработка отсутствующих полей в JSON, валидация формата дат. Основные точки сбоя: недоступность API, сетевые ошибки, проблемы подключения к MinIO или ClickHouse. Ошибки обрабатываются на каждом этапе без полной остановки пайплайна.

2. Развертывание системы

Создание и запуск контейнеров. Все компоненты развертываются через Docker Compose. Файл docker-compose.yaml описывает 4 сервиса: MinIO (объектное хранилище), ClickHouse (база данных), Prefect Server (оркестратор) и Prefect Agent (исполнитель задач). После команды docker-compose up -d

сервисы запускаются в изолированных контейнерах с автоматической инициализацией: MinIO создает бакет raw-weather, ClickHouse выполняет скрипт init_database.sql для создания таблиц.

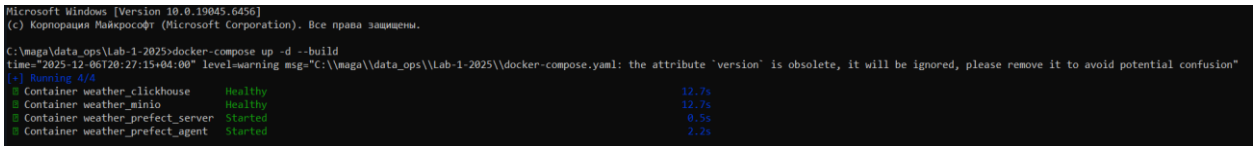


Рисунок 2 – Запуск контейнеров (терминал)

Настройка Telegram бота. Через BotFather в Telegram создан новый бот командой /newbot. После указания имени и username бота система генерирует уникальный API-токен. ID чата получен через бота @webhelpiebot.

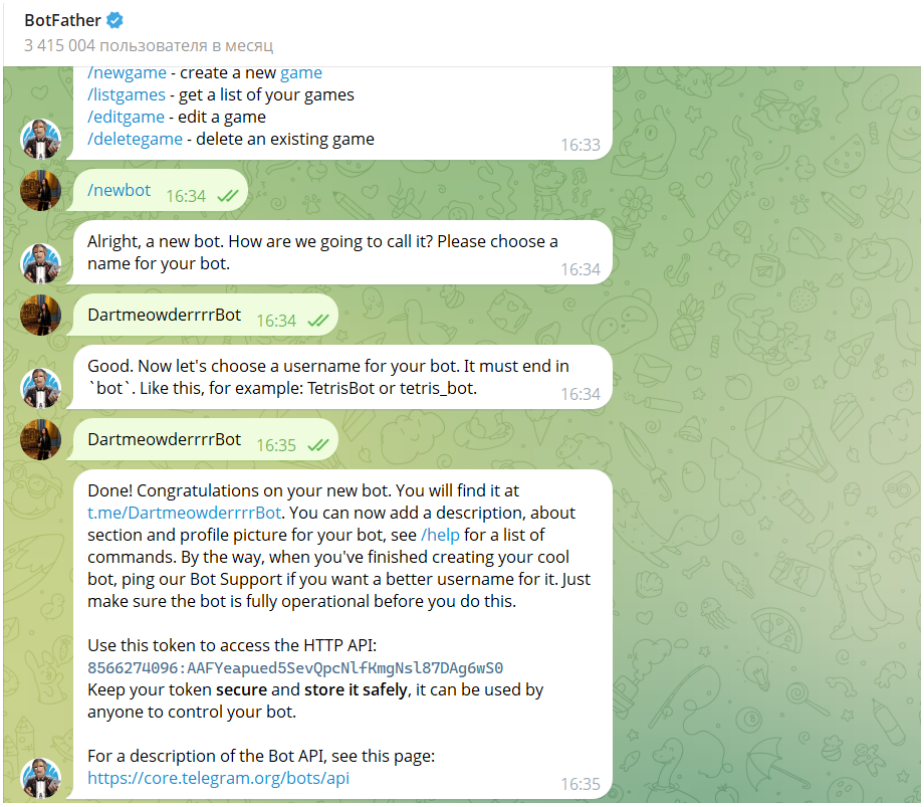


Рисунок 3 – Создание Telegram бота

Токен и ID чата добавляются в .env файл как TELEGRAM_BOT_TOKEN и TELEGRAM_CHAT_ID. Бот настроен на прием сообщений и может отправлять уведомления.

Запуск пайплайна. Flow активируется через Prefect UI или командой в контейнере агента. Prefect Agent подключается к серверу, получает задание и

последовательно выполняет все этапы ETL. После завершения загрузки данных в ClickHouse автоматически отправляется уведомление в Telegram.

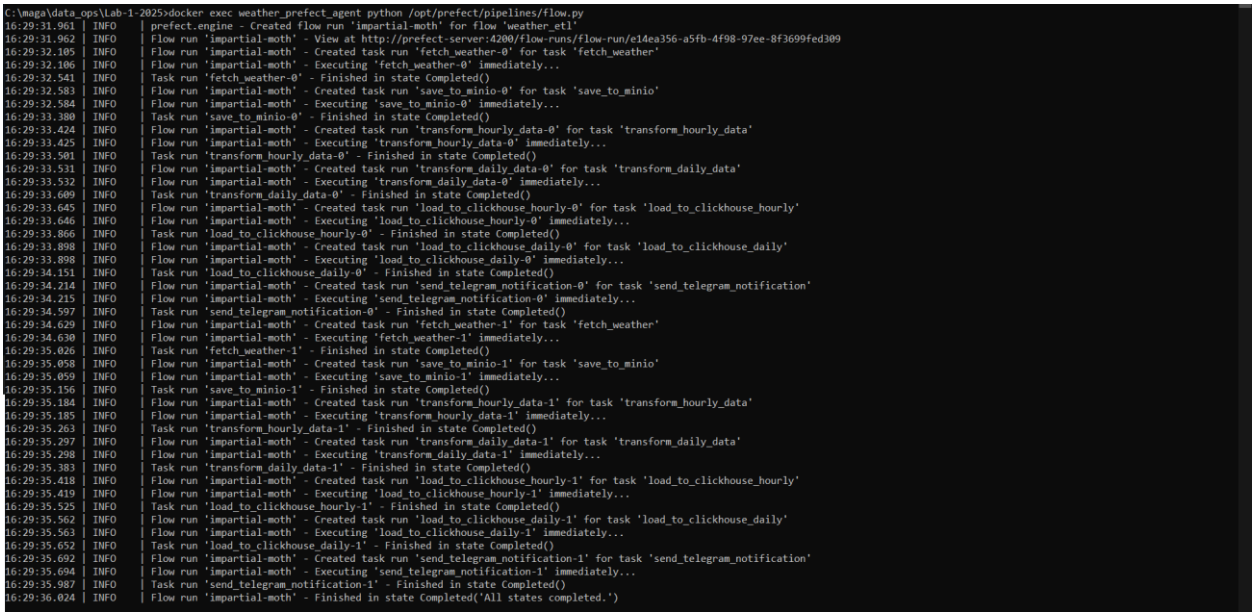


Рисунок 4 – Запуск пайплайна

3. Результаты выполнения

Prefect UI и логи. В веб-интерфейсе Prefect отображаются успешные запуски flow weather_etl с детальным логом каждого таска. Логи содержат время выполнения, статус и сообщения о возможных ошибках.

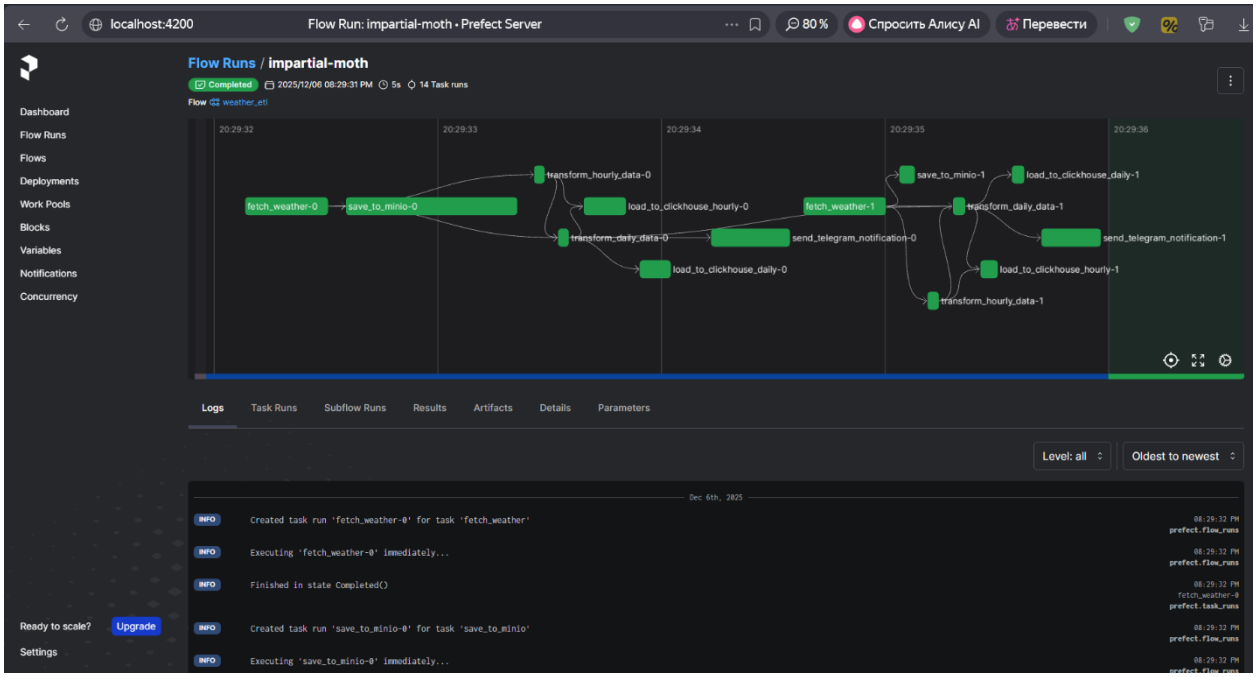


Рисунок 5 – Prefect UI

MinIO bucket. В бакете raw-weather создаются структурированные папки по городам:

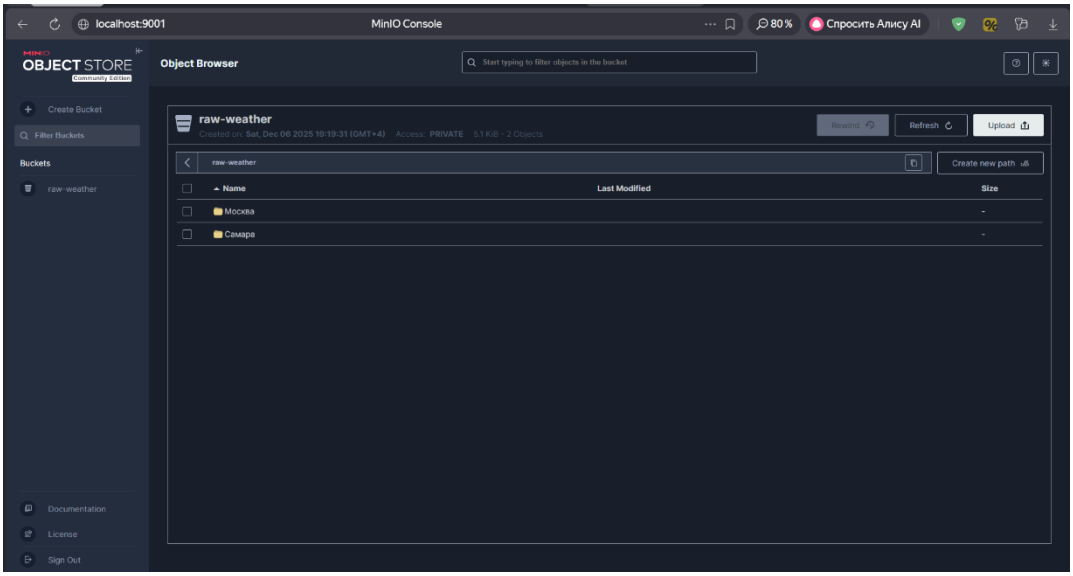
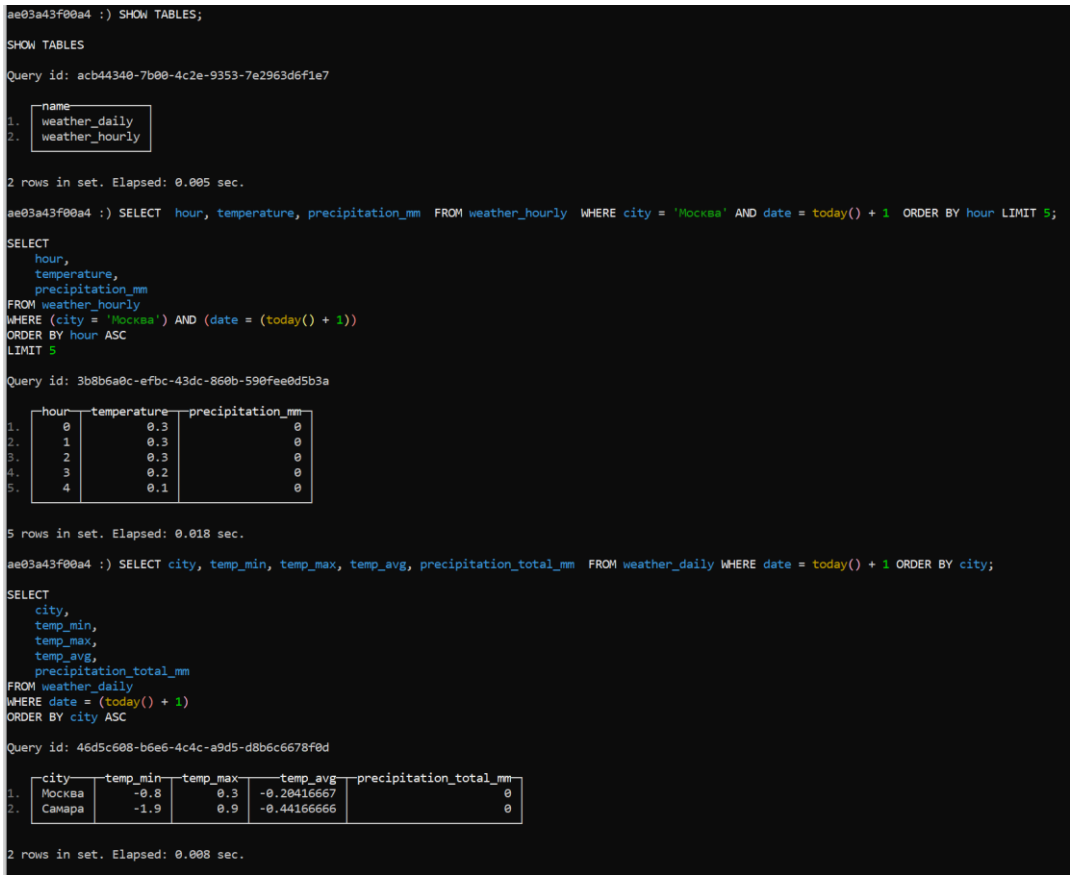


Рисунок 6 – MinIO UI

Каждый JSON-файл содержит полный 48-часовой прогноз с добавленными метаданными city и fetched_at.

ClickHouse данные. Выполнены тестовые запросы к таблицам:



```
ae03a43f00a4 :) SHOW TABLES;

SHOW TABLES

Query id: acb44340-7b00-4c2e-9353-7e2963d6f1e7

   ┌─name─────────┐
1. │ weather_daily  │
2. │ weather_hourly │
   └──────────────┘

2 rows in set. Elapsed: 0.005 sec.

ae03a43f00a4 :) SELECT  hour, temperature, precipitation_mm  FROM weather_hourly  WHERE city = 'Москва' AND date = today() + 1  ORDER BY hour LIMIT 5;

SELECT
    hour,
    temperature,
    precipitation_mm
FROM weather_hourly
WHERE (city = 'Москва') AND (date = (today() + 1))
ORDER BY hour ASC
LIMIT 5

Query id: 3b8b6a0c-efbc-43dc-860b-590fee0d5b3a

   ┌─hour─┬─temperature─┬─precipitation_mm─┐
1. │    0 │         0.3 │                0 │
2. │    1 │         0.3 │                0 │
3. │    2 │         0.3 │                0 │
4. │    3 │         0.2 │                0 │
5. │    4 │         0.1 │                0 │
   └──────┴─────────────┴──────────────────┘

5 rows in set. Elapsed: 0.018 sec.

ae03a43f00a4 :) SELECT city, temp_min, temp_max, temp_avg, precipitation_total_mm  FROM weather_daily WHERE date = today() + 1 ORDER BY city;

SELECT
    city,
    temp_min,
    temp_max,
    temp_avg,
    precipitation_total_mm
FROM weather_daily
WHERE date = (today() + 1)
ORDER BY city ASC

Query id: 46d5c608-b6e6-4c4c-a9d5-d8b6c6678f0d

   ┌─city───┬─temp_min─┬─temp_max─┬─────temp_avg─┬─precipitation_total_mm─┐
1. │ Москва │     -0.8 │      0.3 │ -0.20416667  │                      0 │
2. │ Самара │     -1.9 │      0.9 │ -0.44166666  │                      0 │
   └────────┴──────────┴──────────┴──────────────┴────────────────────────┘

2 rows in set. Elapsed: 0.008 sec.
```

Рисунок 7 – Запросы к таблицам

Telegram уведомления. Модуль отправляет прогноз с температурой (мин/макс/ср) и осадками, добавляет предупреждения и использует Markdown-разметку для читаемости.

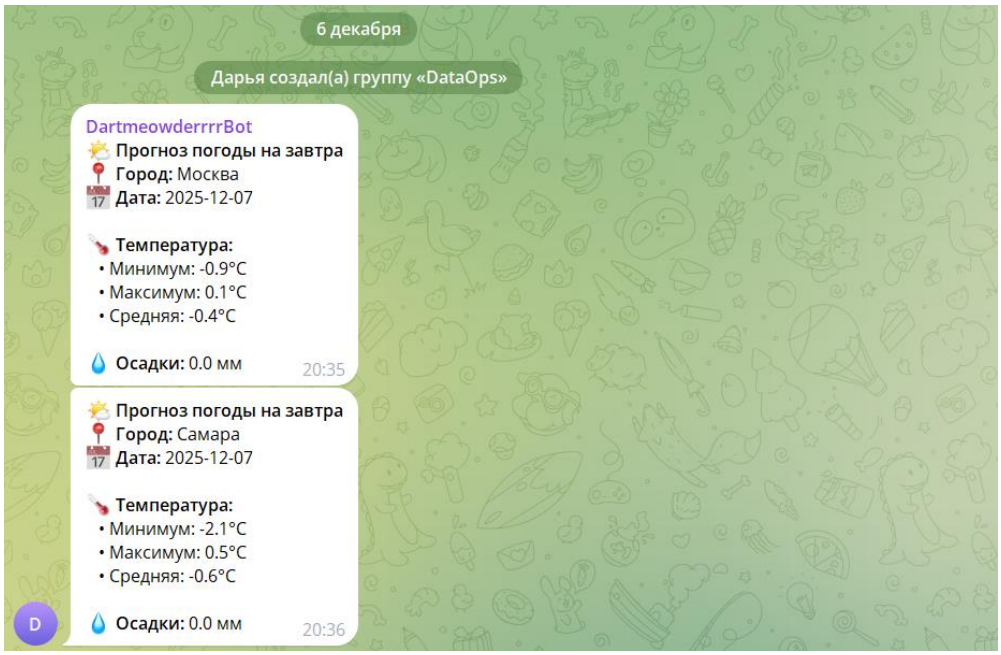


Рисунок 8 – Telegram уведомления в чате

ЗАКЛЮЧЕНИЕ

В результате выполнения лабораторной работы был успешно реализован ETL-пайплайн для сбора, обработки и хранения прогнозов погоды. Система автоматически извлекает данные из Open-Meteo API для Москвы и Самары, сохраняет сырые JSON в MinIO, преобразует их в структурированный формат, загружает в аналитическую СУБД ClickHouse и отправляет уведомления через Telegram.

Основные сложности были связаны с синхронизацией временных интервалов и отладкой взаимодействия между контейнеризованными сервисами.