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»

Выполнил: Миролюбов В.Б.

Группа: 6233-010402D

Самара 2025

Кратко об архитектуре: схема пайплайна, какие инструменты и почему.

Схема пайплайна:

Prefect (оркестратор) → Open-Meteo API (источник) → MinIO (сырые данные) → ClickHouse (агрегированные таблицы) → Telegram Bot (уведомления).

Какие инструменты и почему:

- Prefect 3.x — для оркестрации и расписания (гибкость, UI, управление зависимостями).

- MinIO — S3-совместимое хранилище для сырых JSON-ответов (дешево, просто, локально).

- ClickHouse — колоночная СУБД для аналитики (быстрые агрегации, масштабируемость).

- Telegram Bot API — для push-уведомлений (удобно, бесплатно, современно).

Источник данных: эндпоинт Open-Meteo, параметры запроса.

Используется бесплатный API Open-Meteo (https://open-meteo.com/).

Эндпоинт: https://api.open-meteo.com/v1/forecast

Параметры запроса:

- latitude, longitude — координаты Москвы и Самары, определены примерно.

- hourly=temperature_2m, precipitation, wind_speed_10m, wind_direction_10m — нужные метрики для анализа.

- forecast_days=2 — чтобы гарантированно получить завтрашний день.

- timezone=Europe/Moscow — для корректного времени.

Extract → Transform → Load: по 2–3 предложения на этап.

Extract: Ежедневно в 23:15 Prefect запускает flow, который делает HTTP-запросы к Open-Meteo API для указанных городов. Сырые JSON-ответы сохраняются в MinIO для аудита и отладки.

Transform: Почасовые данные нормализуются в плоскую структуру для таблицы weather_hourly. Для weather_daily рассчитываются агрегаты: min/max/avg температура,

сумма осадков, и генерируются предупреждения при сильном ветре (>15 м/с) или осадках (>10 мм).

Load: Преобразованные данные загружаются в ClickHouse с помощью clickhouse-connect. Отправляется краткое текстовое уведомление в Telegram с прогнозом и алертами.

Качество данных: какие проверки качества данных реализованы и возможные точки сбоя.

Проверки:

- Валидация HTTP-статуса (200) при запросе к API.

- Проверка наличия координат для города.

- Проверка, что для завтрашнего дня есть хотя бы одна почасовая запись.

- Обработка None значений при нормализации.

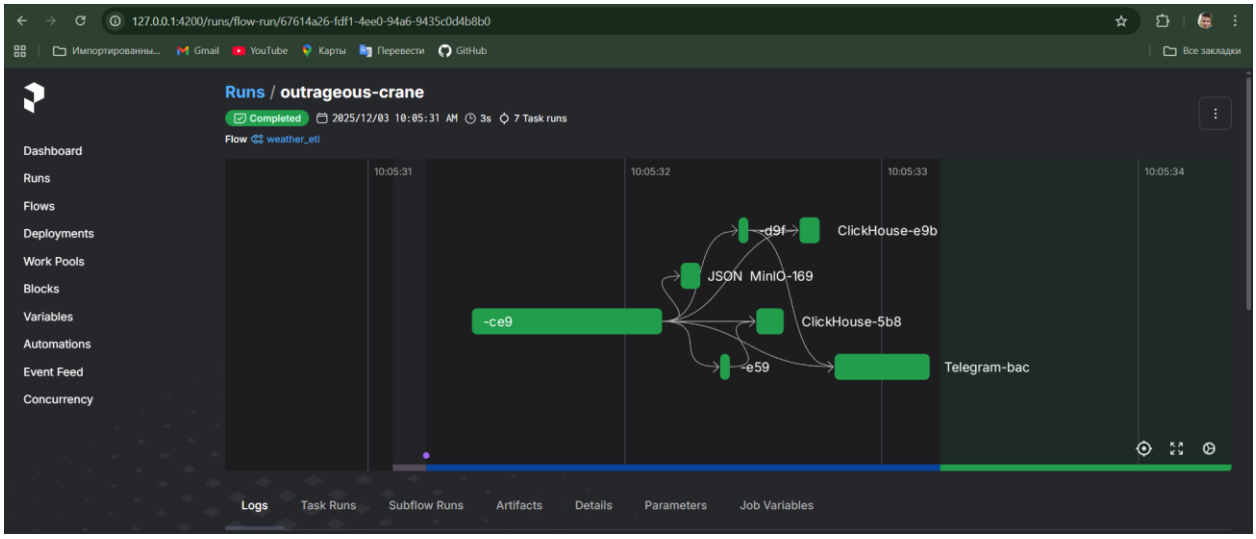
Точки сбоя:

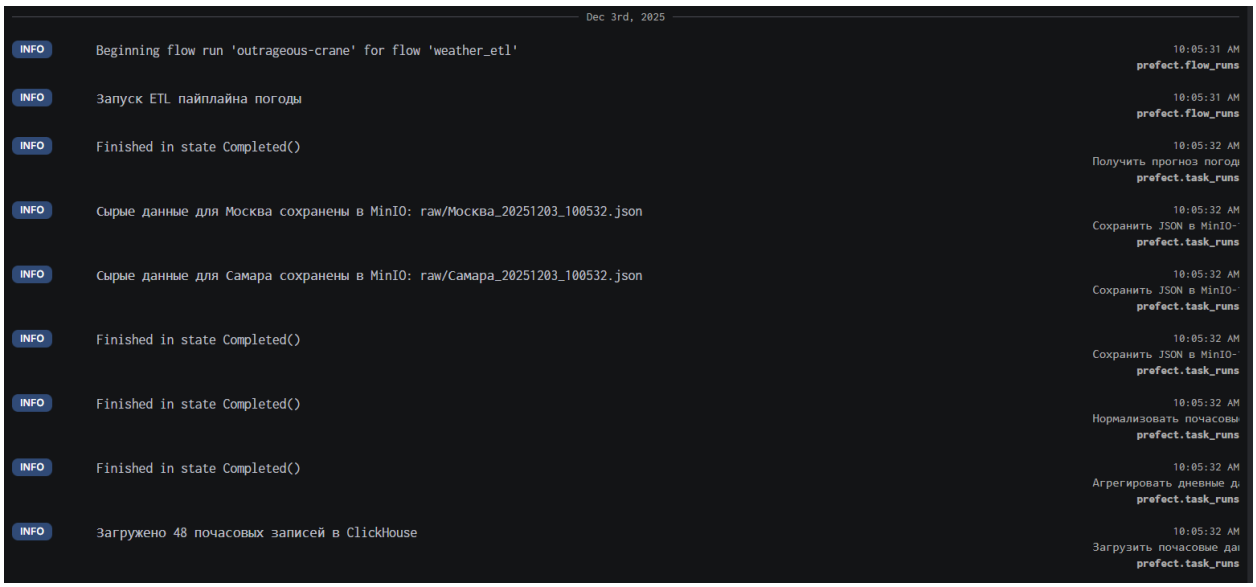
- Недоступность Open-Meteo API (обрабатывается через retries=3 в Prefect).

- Неверные/отсутствующие секреты (MinIO, CH, Telegram) — хранятся в Prefect Blocks.

Результаты работы пайплайна:

1. Логи Prefect





Рисунки 1, 2, 3 – Логи работы Prefect

2. Содержимое бакета в MinIO

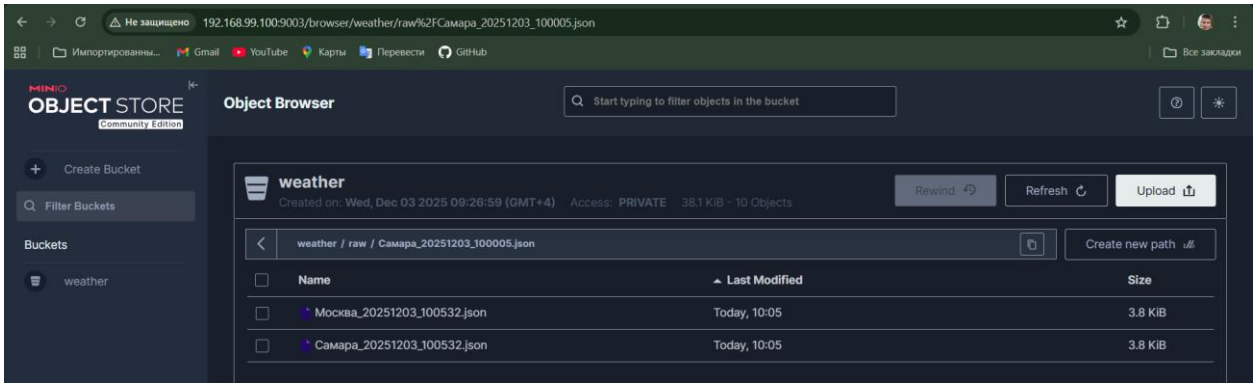


Рисунок 4 – Логи работы Prefect

3. Содержимое в Click

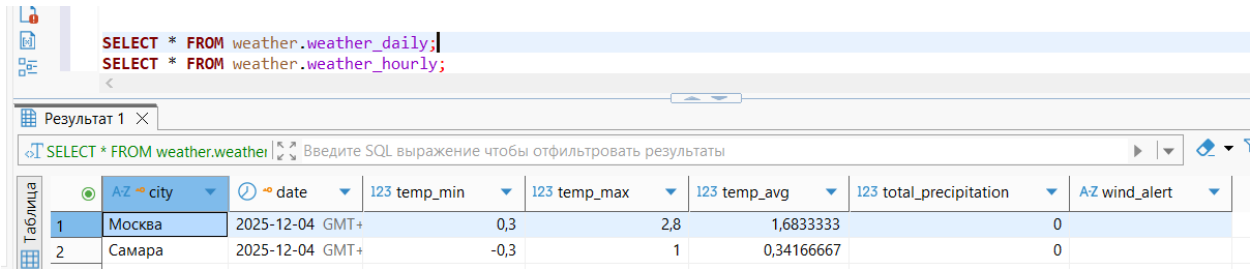


Рисунок 5 – Содержимое таблицы weather_daily

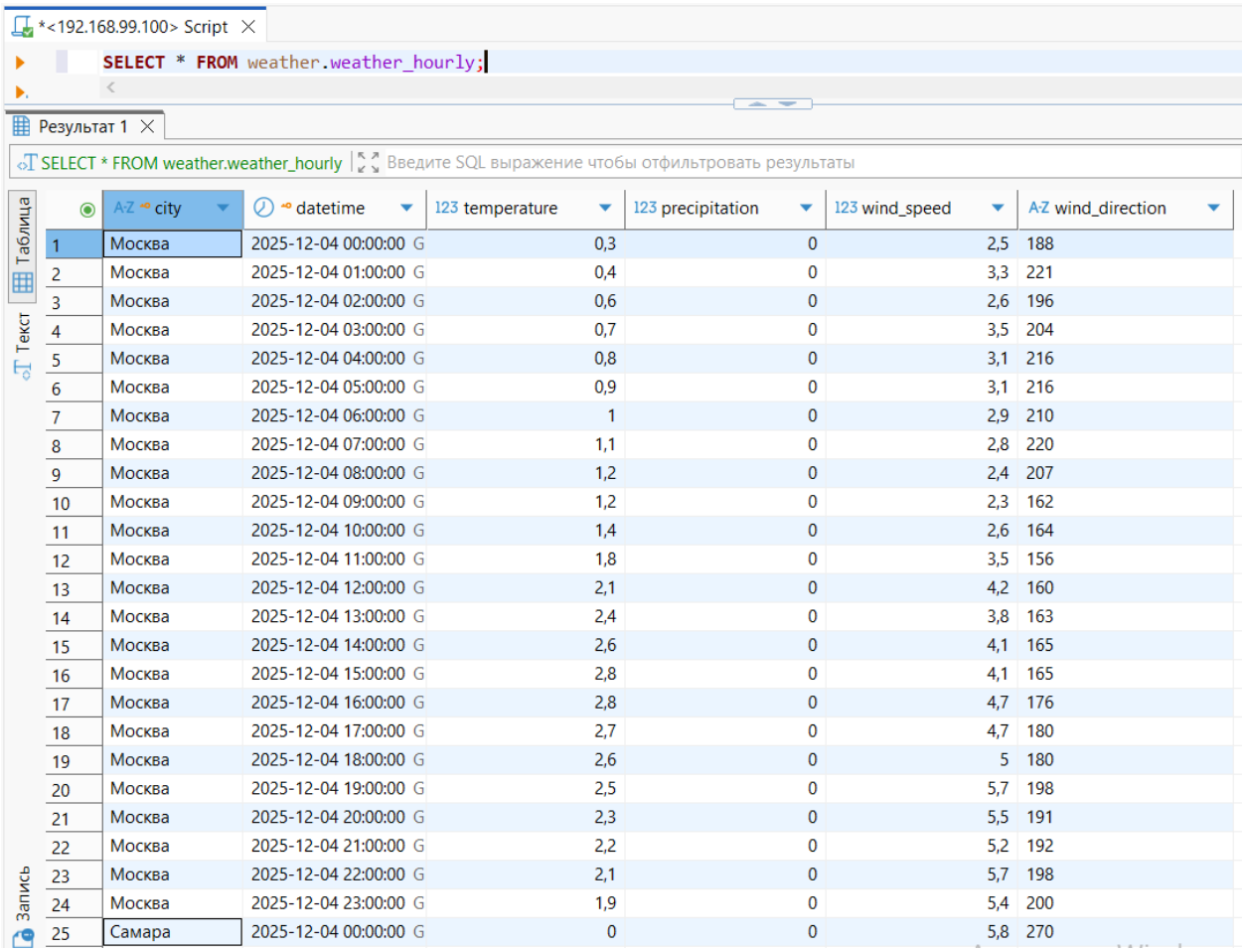


Рисунок 6 – Содержимое таблицы weather_hourly

4. Уведомления в Telegram

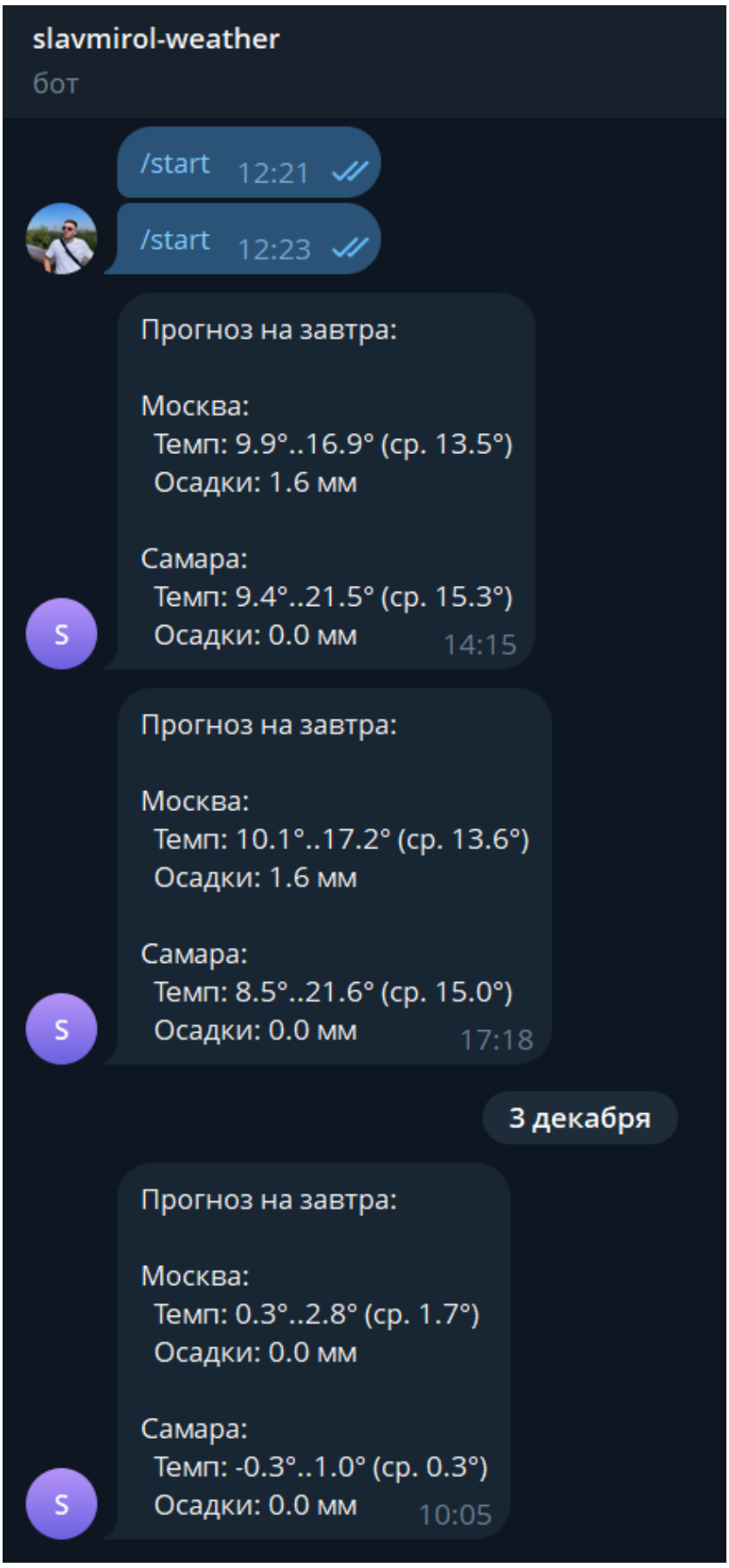


Рисунок 7 – Уведомления в телеграм

Выводы: что было сложным, что бы улучшили.

Сложным было разобраться со структурой Prefect: сервер, деплой, work-pool, процесс.

Что бы улучшил?

- Реализовал бы алерты при сбое flow (Slack/email).

- Добавил бы Grafana-дашборд для визуализации.