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 №3

Дисциплина: «Проектирование программных комплексов»

Дисциплина: «Инженерия данных»

Выполнил: Седых М.Ю.
Группа: 6232-010402D

Самара 2025

1.1 Введение

Цель работы — реализовать полный цикл оркестрации ML-проекта для прогнозирования погоды: от загрузки исторических данных до развертывания инференс-сервиса и автоматической отправки прогнозов. Проект включает создание ClearML Dataset, обучение моделей с логированием экспериментов, поиск оптимальных гиперпараметров (HPO), регистрацию лучшей модели в Model Registry, развертывание FastAPI сервиса для прогноза на 7 дней вперед и настройку n8n-пайплайна для еженедельной автоматической отправки прогнозов в Telegram.

1.2 Описание данных и источников

Источник данных:

- API: Open-Meteo Historical Weather API
- URL: https://archive-api.open-meteo.com/v1/archive
- Тип: публичный API без ключа
- Период: последние 3+ года (с 2021 года по настоящее время)
- География: Москва (55.7558°N, 37.6173°E) и Санкт-Петербург (59.9343°N, 30.3351°E)

Целевая переменная:

- `temp_avg` — средняя дневная температура (°C)

Входные признаки:

- Календарные: день недели, день года, месяц, год, синусоидальные признаки сезонности
- Лаги температуры: temp_avg, temp_max, temp_min за 1, 2, 3, 7, 14 дней назад
- Лаги осадков: precipitation_sum за 1, 2, 3, 7 дней назад
- Скользящие агрегаты: средние/максимальные/минимальные температуры за 7, 14, 30 дней; суммы осадков за те же периоды
- Производные: разница температур, изменение температуры за 1 и 7 дней, индикаторы осадков

Выходные данные:

- Прогнозы средней температуры на 7 дней вперед
- Доверительные интервалы для каждого прогноза
- JSON-ответ с метаданными

1.3 Архитектура и методы

1.3.1 Общая архитектура системы

Система состоит из следующих компонентов:

1) ClearML Server — сервер для логирования экспериментов, управления датасетами и моделями (MongoDB, Elasticsearch, Redis)

2) Data Loader — скрипт загрузки исторических данных и создания ClearML Dataset

3) Training Service — скрипт обучения моделей с логированием в ClearML

4) HPO Service — скрипт поиска гиперпараметров (Grid Search, минимум 10 конфигураций)

5) Inference Service — FastAPI сервис для прогнозирования (порт 8000)

6) n8n — оркестрация еженедельного пайплайна с отправкой в Telegram

Все компоненты развернуты через Docker Compose и взаимодействуют через HTTP API и ClearML SDK.

1.3.2 Технологический стек

Версии используемых инструментов:

- Python: 3.10-slim
- ClearML Server: latest (allegroai/clearml)
- MongoDB: 6.0
- Elasticsearch: 8.13.4
- Redis: 8.4.0
- LightGBM: >=4.0.0
- FastAPI: >=0.104.0

- Uvicorn: >=0.24.0
- n8n: latest
- pandas: >=2.0.0
- numpy: >=1.24.0
- scikit-learn: >=1.3.0

1.3.3 Структура репозитория

Репозиторий содержит конфигурацию и код для системы прогнозирования погоды. Включает docker-compose.yml и Dockerfile для сервисов, requirements.txt для зависимостей, шаблон .env и clearml.conf. В папке src находятся скрипты для загрузки данных, обучения моделей, поиска гиперпараметров и FastAPI сервиса для прогнозов. Каталоги data и models хранят датасеты и обученные модели, workflows содержит экспортированный n8n workflow.

1.3.4 Описание пайплайна

Этап 1: Загрузка данных (data_loader.py)

Скрипт загружает исторические данные из Open-Meteo API для выбранных городов, разбивая период по годам с retry-логикой. Создаются признаки и целевые переменные для горизонтов D+1...D+7. Датасет сохраняется локально и загружается в ClearML Dataset с метаданными и статистикой.

Этап 2: Обучение модели (train.py)

Скрипт обучает отдельные модели LightGBM для каждого горизонта прогноза (7 моделей) с временным разбиением данных (80% train, 20% test). Метрики (MAE, RMSE, MAPE), графики важности признаков и scatter plots логируются в ClearML. Ансамбль моделей сохраняется в pickle-файл и регистрируется в ClearML Model Registry.

Этап 3: Поиск гиперпараметров (hpo.py)

Выполняется Grid Search по сетке гиперпараметров (минимум 10 конфигураций) с Time Series Cross-Validation (3 фолда). Результаты всех

экспериментов логируются в ClearML. Лучшая модель обучается на всех данных и регистрируется в Model Registry с тегами и версией.

Этап 4: Инференс-сервис (inference_service.py)

FastAPI сервис загружает лучшую модель при старте. Эндпоинт /predict принимает город и список дат, получает исторические данные через Open-Meteo API, создает признаки и возвращает прогнозы с доверительными интервалами. Дополнительно предоставляются эндпоинты /health и /cities.

Этап 5: n8n пайплайн (weather_forecast_workflow.json)

Workflow запускается каждую неделю в понедельник в 07:00 по расписанию cron. Выполняется health check inference-сервиса, параллельно отправляются запросы на прогноз для Москвы и Санкт-Петербурга. Результаты объединяются, форматируются в читаемое сообщение и отправляются в Telegram. При ошибках отправляется уведомление об ошибке.

1.3.5 Реализация компонентов

auto_subtitle_service.py реализует HTTP API на Flask для распознавания речи с использованием Whisper. Сервис предоставляет эндпоинты для проверки работоспособности и для распознавания речи с возвратом результата в формате JSON или SRT. Модель Whisper загружается при первом запросе и кэшируется, используется английский язык распознавания, временные метки формируются в формате HH:MM:SS,mmm. Поддерживается выбор размера модели и устройства выполнения.

video_processing_service.py представляет собой HTTP API на Flask для обработки видео с помощью FFmpeg. Сервис выполняет извлечение аудио из видеофайлов с приведением к формату WAV 16 kHz mono PCM и встраивание SRT-субтитров в видео с кодированием H.264. Реализованы проверки входных файлов, базовая валидация SRT, контроль прав доступа и автоматическая очистка временных файлов.

start_services.sh — Bash-скрипт, запускающий оба Python-сервиса в одном контейнере на разных портах и обеспечивающий корректное завершение процессов при получении сигналов SIGTERM и SIGINT.

1.3.6 Конфигурация Docker

docker-compose.yml описывает сервисы:

1) ClearML Server с apiserver, webserver, fileserver, mongo, elasticsearch, redis
2) inference-service на FastAPI с healthcheck
3) training-service для обучения и HPO
4) n8n для оркестрации с доступом к локальному inference-service через host.docker.internal

Dockerfile реализует multi-stage сборку:

1) base: Python 3.10-slim с curl и git
2) training-service: зависимости, код и данные
3) inference-service: зависимости, код и модели, порт 8000
4) data-loader: минимальный образ для загрузки данных
5) hpo-service: образ для запуска HPO

1.3.7 Обработка ошибок

В data_loader используется retry-логика для API-запросов с экспоненциальной задержкой. В inference_service предусмотрена заглушка (stub) для прогноза при недоступности модели, используется fallback на климатические данные. В n8n workflow реализованы проверки health check, обработка ошибок с отправкой уведомлений в Telegram. На уровне Docker применяются healthcheck для всех сервисов, управляемые зависимости через `depends_on` и автоматический перезапуск контейнеров.

1.4 Эксперименты и результаты

Было загружено более 2270 записей для двух городов за примерно три года, создано более 50 признаков и датасет сохранён в ClearML. Обучено семь моделей LightGBM для горизонтов D+1…D+7 с временным разбиением 80/20, метрики логируются в ClearML. Для поиска гиперпараметров выполнен Grid Search по 10 конфигурациям с Time Series Cross-Validation, лучшие параметры достигли RMSE 2.79°C и MAE 2.11°C, модель зарегистрирована в ClearML Model Registry.

FastAPI сервис inference развернут на порту 8000, обрабатывает запросы менее чем за секунду, поддерживает Москву и Санкт-Петербург, возвращает прогнозы на 7 дней с доверительными интервалами. n8n пайплайн запускается по cron в понедельник утром, автоматически отправляет прогнозы в Telegram с форматированием сообщений и уведомлениями об ошибках. Время загрузки данных составляет 2–5 минут, обучение 5–10 минут, если базовая модель или 15–30 минут (HPO), используется 2–4 GB RAM для обучения и ~500 MB для inference, дисковое место около 150 MB.

1.5 Обсуждение

Система обеспечивает полный цикл ML-проекта: от данных до деплоя. ClearML интегрирован на всех этапах для логирования экспериментов и управления моделями. Модульная архитектура позволяет легко изменять компоненты. Ограничения: зависимость от внешнего API Open-Meteo, необходимость запуска ClearML Server, 2 города. Возможные улучшения: добавление больше городов, использование дополнительных источников данных, расширение признаков (погодные индексы, внешние данные), эксперименты с другими моделями (XGBoost, CatBoost, LSTM).

1.6 Выводы

Реализован полный цикл ML-проекта для прогнозирования погоды с использованием ClearML, FastAPI и n8n. Создан ClearML Dataset с историческими данными за 3+ года для 2 городов. Реализовано обучение ансамбля моделей LightGBM с логированием всех метрик и артефактов в ClearML. Выполнен поиск гиперпараметров с использованием Time Series Cross-Validation, лучшая модель зарегистрирована в Model Registry. Развернут FastAPI инференс-сервис для прогнозирования на 7 дней с доверительными интервалами. Настроен n8n пайплайн для еженедельной автоматической отправки прогнозов в Telegram. Все компоненты упакованы в Docker и воспроизводимы через docker-compose.

1.7 Ссылки на документацию

- ClearML: https://clear.ml/docs
- Open-Meteo API: https://open-meteo.com/en/docs/historical-weather-api
- LightGBM: https://lightgbm.readthedocs.io/
- FastAPI: https://fastapi.tiangolo.com/
- n8n: https://docs.n8n.io/
- Docker Compose: https://docs.docker.com/compose/
- scikit-learn: https://scikit-learn.org/stable/

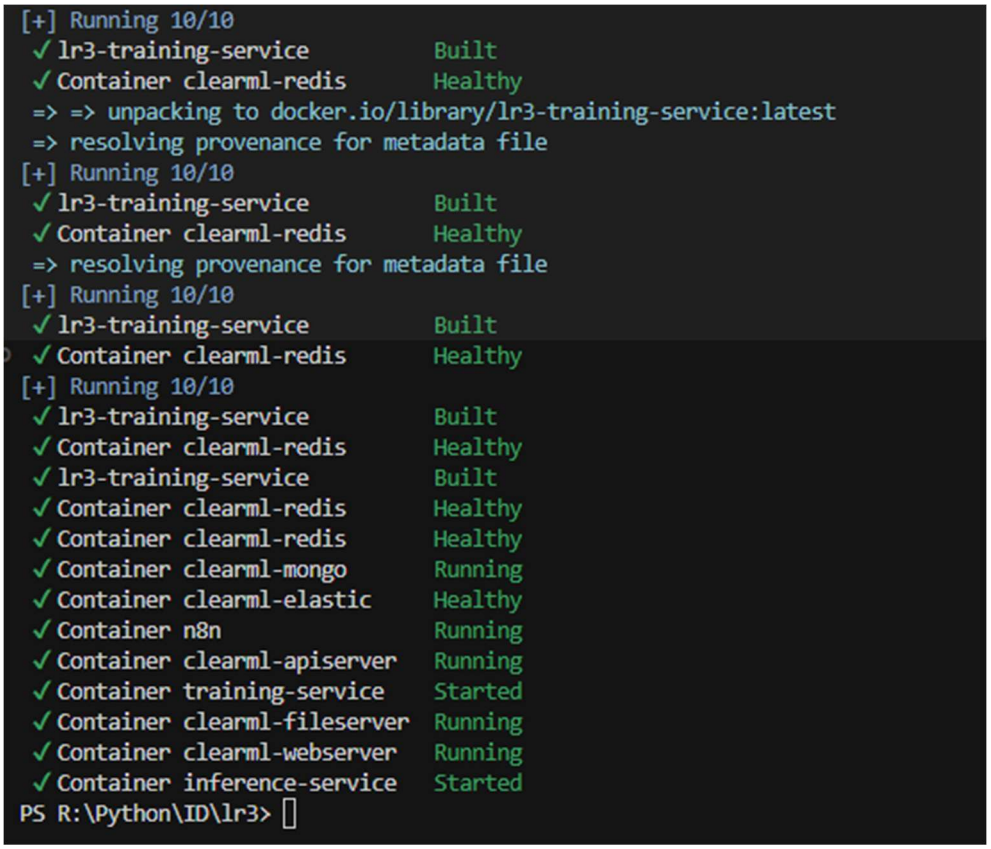

Рисунок 1 - Статус всех Docker-контейнеров системы

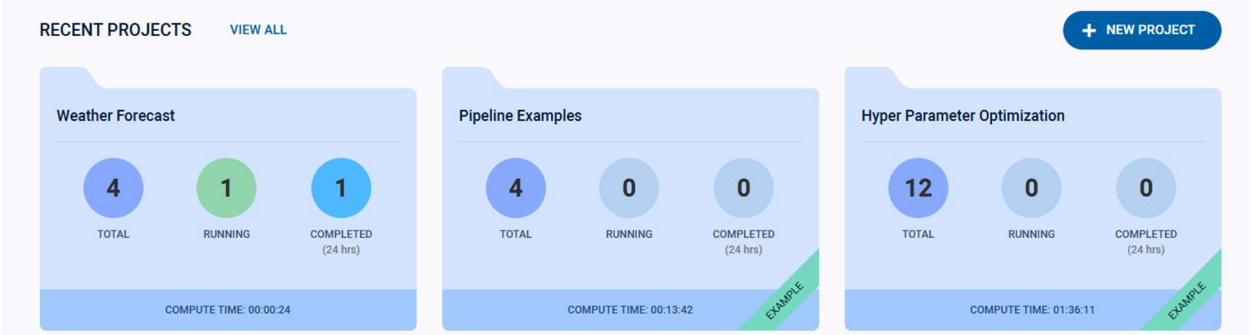

Рисунок 2 – Project ClearML

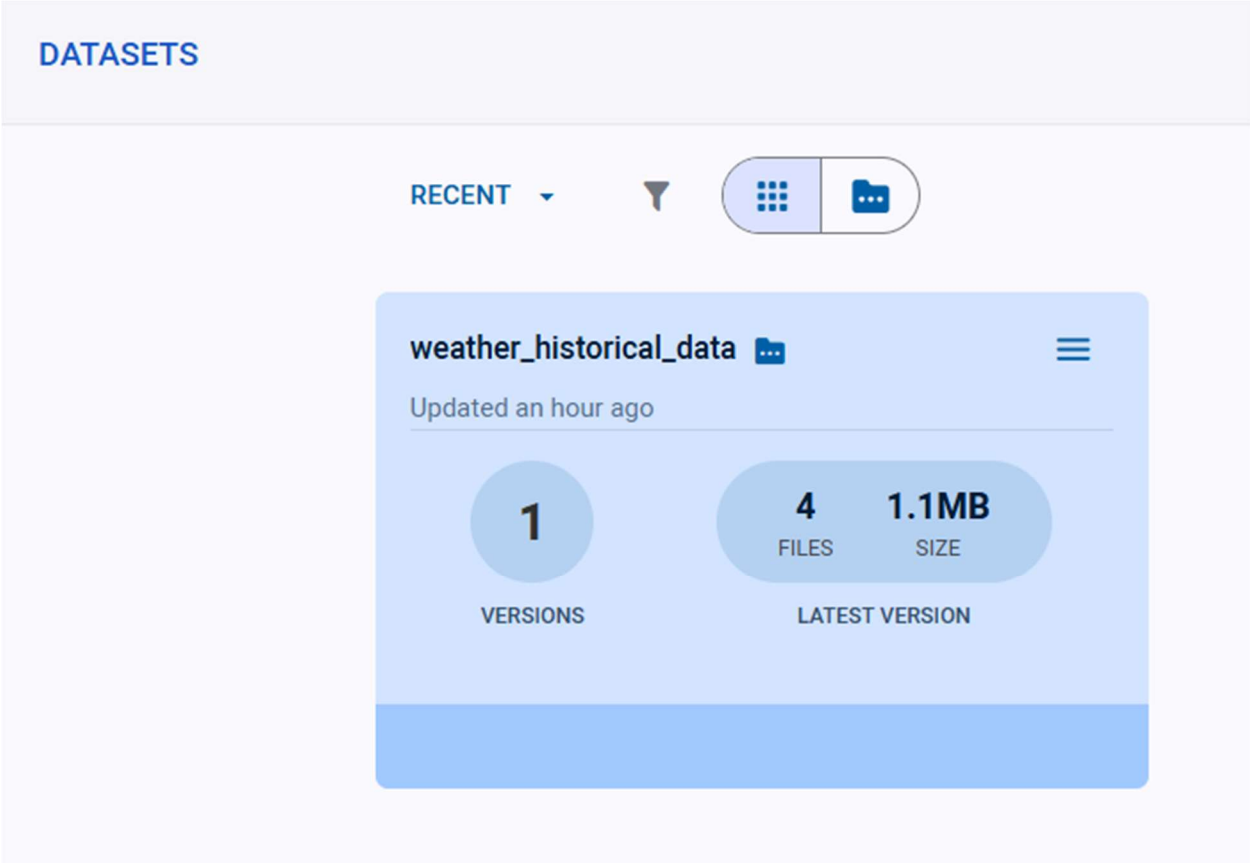

Рисунок 3 - ClearML Dataset

Таблица 1 - Сводка метрик по горизонтам

Горизонт	**MAE**	**RMSE**	**MAPE, %**
D+1	1.64	2.07	43.66
D+2	2.23	2.79	44.80
D+3	2.48	3.12	44.65
D+4	2.51	3.18	40.62
D+5	2.74	3.41	52.12
D+6	2.82	3.48	52.34
D+7	2.87	3.51	61.21

Рисунок 3 – Вывод в ClearML после HPO

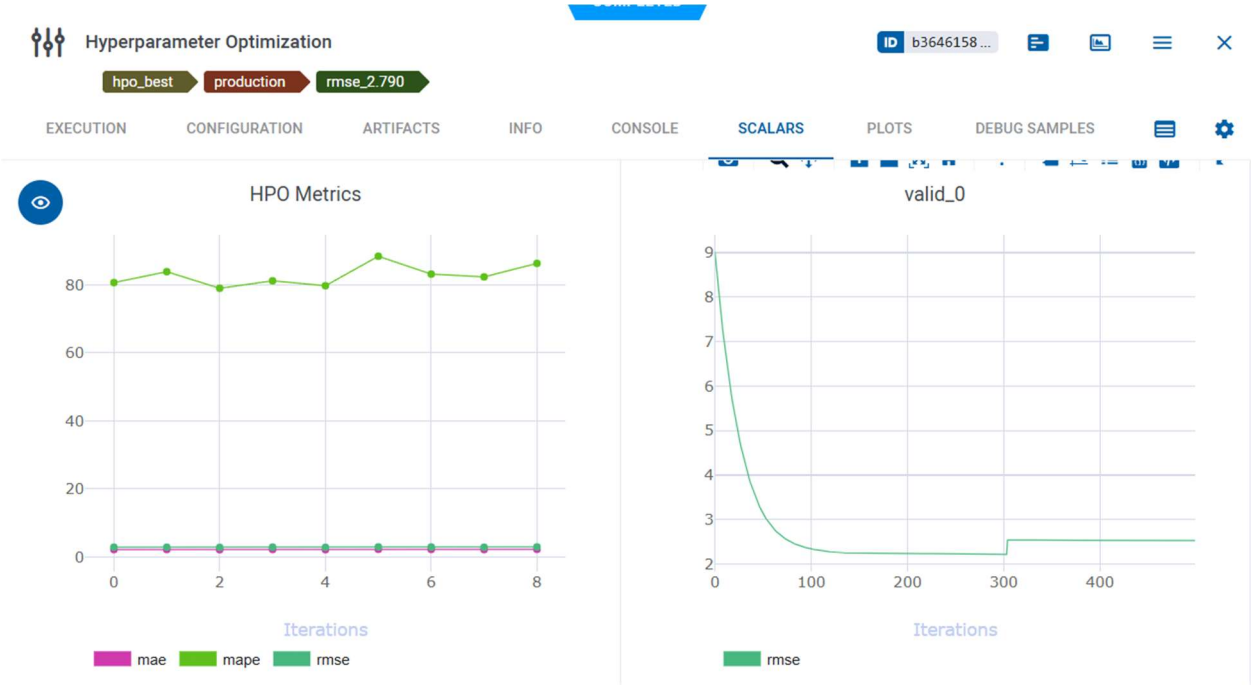

Рисунок 4 – График метрик

Таблица 2 – Лучшие параметры и метрики

Параметр / Метрика	**Значение**
n_estimators	500
early_stopping_rounds	50
num_leaves	15
learning_rate	0.1
feature_fraction	0.8
bagging_fraction	0.8
RMSE	2.7905
MAE	2.1083
MAPE, %	78.94

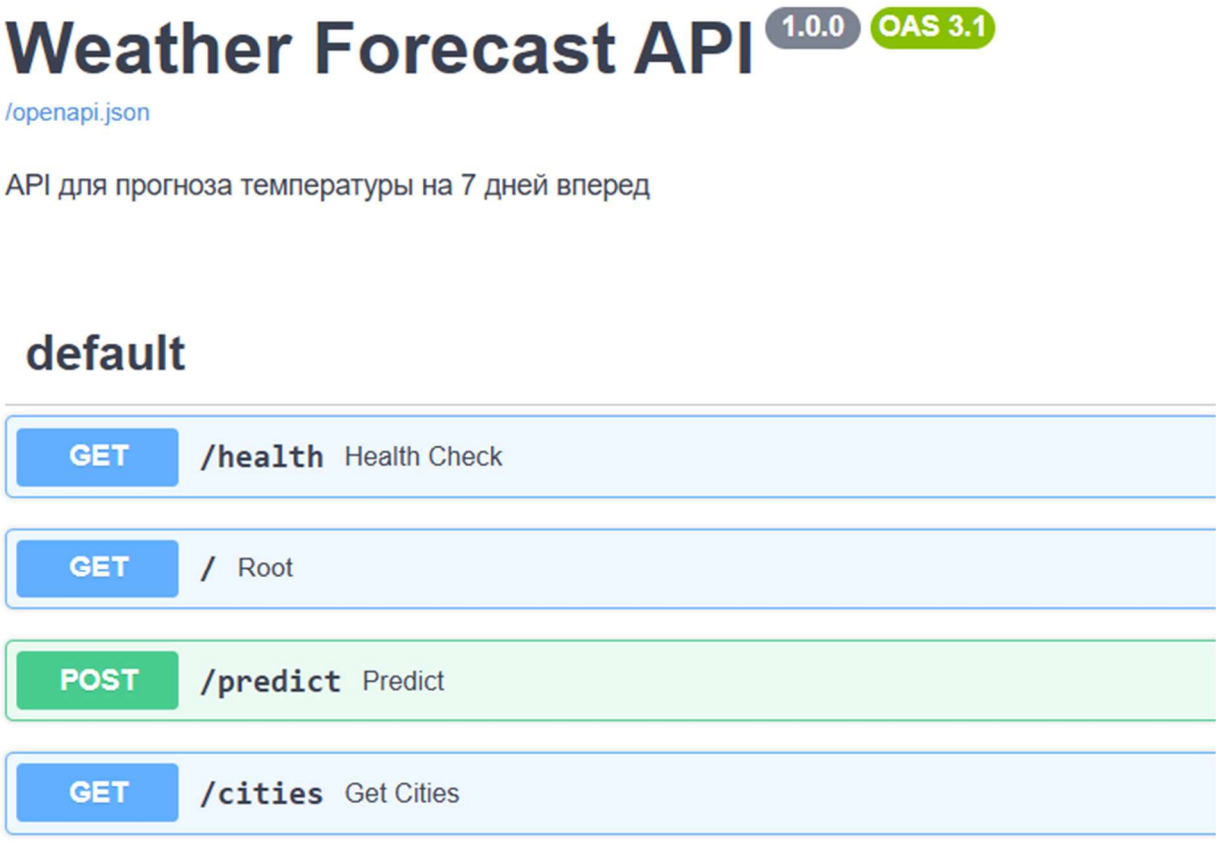

Рисунок 5 - Swagger UI

Curl

```
curl -X 'POST' \
  'http://localhost:8000/predict' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "city": "Москва",
  "dates": [
    "2025-12-16",
    "2025-12-17",
    "2025-12-18",
    "2025-12-19",
    "2025-12-20",
    "2025-12-21",
    "2025-12-22"
  ],
  "additional_features": {}
}'
```

Request URL

```
http://localhost:8000/predict
```

Server response

Code	Details
200	Response body


```
{
  "city": "Москва",
  "request_date": "2025-12-15",
  "forecasts": [
    {
      "date": "2025-12-16",
      "day_offset": 1,
      "predicted_temp_avg": -4.9,
      "confidence_interval": {
        "lower": -8.4,
        "upper": -1.4
      },
      "prediction_std": 1.8
    },
    {
      "date": "2025-12-17",
      "day_offset": 2,
      "predicted_temp_avg": -2.8,
      "confidence_interval": {
        "lower": -6.9,
        "upper": 1.3
      },
      "prediction_std": 2.1
    },
    {
      "date": "2025-12-18",
      "day_offset": 3,
      "predicted_temp_avg": -4.9,
```


Рисунок 6 – Ответ API /predict

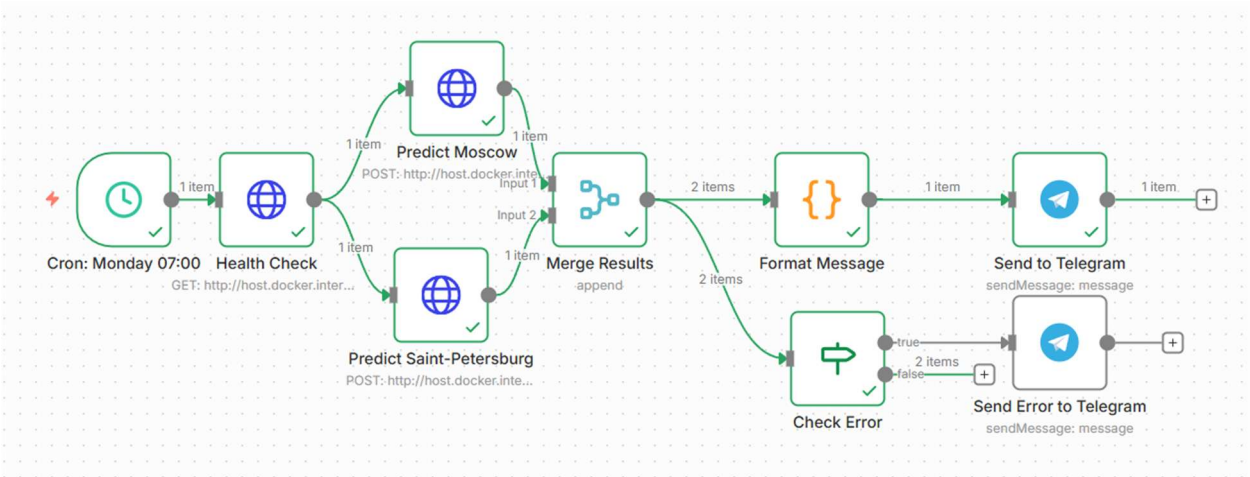

Рисунок 7 - n8n Workflow

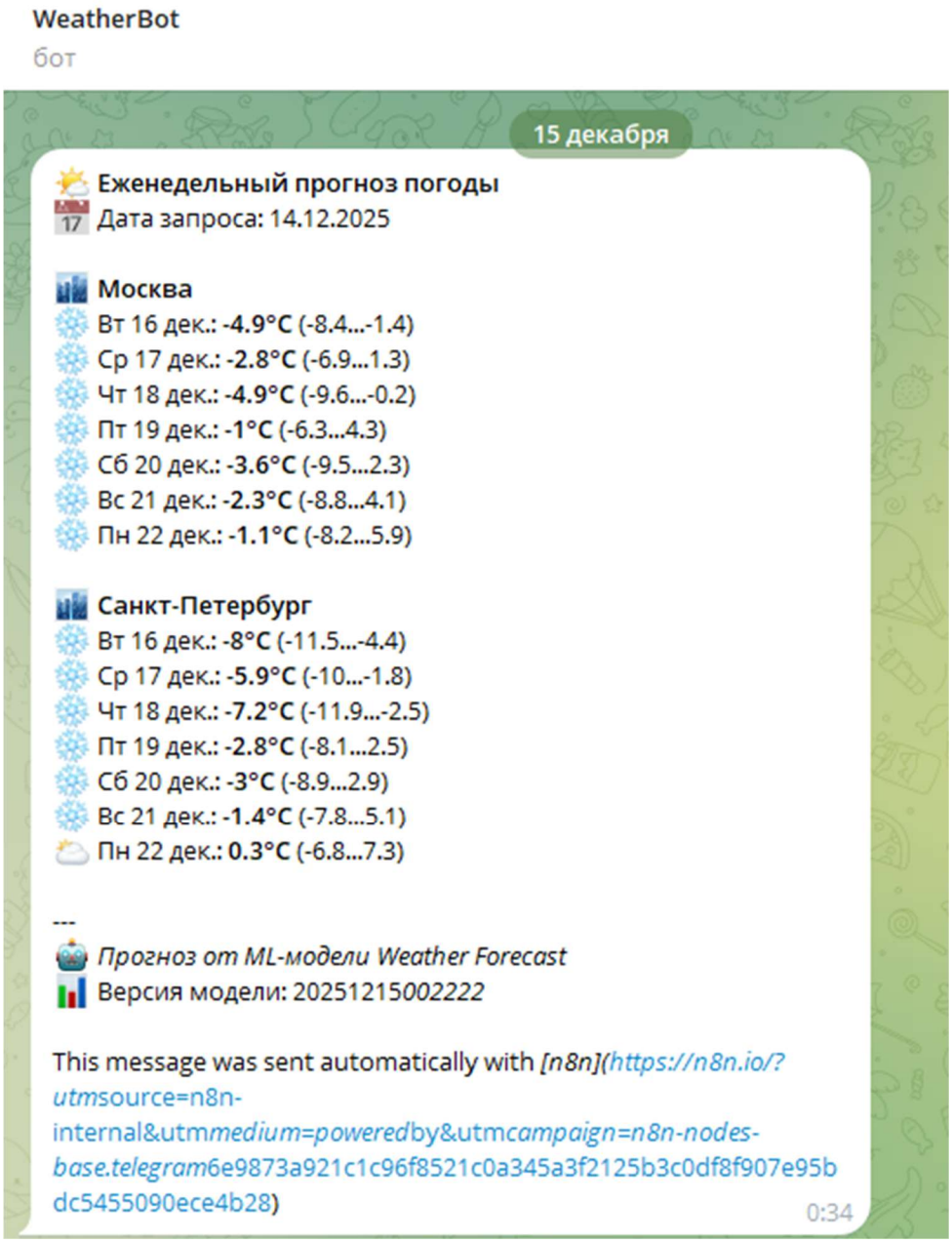

Рисунок 8 - Telegram сообщение