ЛИДЕРЫ ЦИФРОВОЙ ТРАНСФОРМАЦИИ

ПРОЕКТ МЭРА МОСКВЫ

ДЕПАРТАМЕНТ ПРЕДПРИНИМАТЕЛЬСТВА И ИННОВАЦИОННОГО РАЗВИТИЯ ГОРОДА МОСКВЫ

РАЗВИТИЕ ЧЕЛОВЕЧЕСКОГО КАПИТАЛА

ДЕПАРТАМЕНТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ГОРОДА МОСКВЫ

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

ЗАДАЧА 6

Интеллектуальный цифровой инженер данных

1. Актуальность задачи

В условиях стремительного развития цифровой экономики и роста объёмов данных эффективное управление ресурсами становится ключевым фактором успешной реализации государственных и муниципальных задач.

Департамент информационных технологий города Москвы ежедневно работает с большими массивами данных, поступающих из различных источников – от транспортных систем и городских сервисов до социальных платформ.

Для обеспечения прозрачности, оперативности и качества принимаемых решений важно своевременно собирать, обрабатывать, хранить и анализировать эти данные.

Одной из ключевых ролей в этом процессе является инженер данных – специалист, отвечающий за построение и поддержку инфраструктуры обработки данных.

В банках, ритейле, госструктурах и IT-компаниях ощущается дефицит квалифицированных кадров в области data engineering. При этом значительная часть задач, выполняемых инженерами данных, носит рутинный и типовой характер: подключение к источникам, проектирование ETL-процессов, выбор оптимальных решений для хранения и настройка оркестрации.

С развитием технологий искусственного интеллекта появляется возможность автоматизации этих процессов за счёт создания интеллектуальных систем, способных анализировать структуру данных, принимать обоснованные решения по архитектуре хранения и автоматически строить рабочие пайплайны. Такие решения позволяют не только сократить зависимость от ручного труда, но и повысить скорость внедрения новых аналитических сервисов, обеспечить масштабируемость и стандартизацию подходов к работе с данными.

Разработка интеллектуального цифрового инженера данных на базе ИИ соответствует стратегическим целям цифровизации государственного управления. Реализация подобного решения позволит подразделениям, не обладающим специализированными кадрами, эффективно управлять данными и оперативно решать бизнес- и социально значимые задачи.

2. Описание задачи

Разработайте ассистента с применением модели(ей) машинного обучения (LLM), который возьмёт на себя рутинные и базовые функции инженера данных: анализ источников входных данных, анализ входных данных, выбор оптимальной БД и структуры для хранения данных, построение ETL-конвейеров и автоматизация процессов регулярного обновления полученных конвейеров.

Что должен уметь цифровой инженер:

1. Подключаться к разным источникам данных: Файлы (CSV/XML/JSON), БД (PostgreSQL, ClickHouse) — обязательно, рекомендуемо — kafka, hadoop, spark-streaming.
2. Анализировать структуру данных на основе информации об источнике (предоставляется пользователем сервиса), уметь подключаться к источникам информации о данных и предлагать рекомендации оптимального места хранения (например: агрегированные аналитические данные – в ClickHouse, сырые – в HDFS, оперативные – в PostgreSQL).
3. Генерировать DDL-скрипты на основе образца данных, создавать шаблонные рекомендации (например, партицирование по дате — для временных данных, индексы – для часто используемых полей) и предоставлять выбор для пользователя.
4. Создавать пайплайны (можно с использованием готовых операторов внутри Airflow) без сложных трансформаций — обязазательно, рекомендуемо: с вариативностью (простая фильтрация, агрегаты, объединения).
5. Формировать отчет с обоснованием выбора СУБД, структур хранения и ETL-логики: «Я вижу, что данные временные и агрегированные – предлагаю хранить в ClickHouse с партицированием по дате и запускать обновление раз в час через Airflow».
6. Работать по расписанию — обязательно, рекомендуемо — в режиме реального времени
7. Обрабатывать потоковые данные (Kafka) и выполнять пакетную обработку – рекомендуемо.
8. Предоставлять простой UI, где пользователь указывает:
   - откуда брать данные (источник, реквизиты доступа)
   - куда положить (целевая система, реквизиты доступа)
   - и получает визуализацию пайплайна и рекомендации (в том числе по обновлению)
   - возможность редактировать сгенерированные пайплайны перед запуском

Цель: разработать MVP ИИ-ассистента для автоматизации ETL-задач, позволяющего пользователям без специальных знаний настраивать конвейеры через UI.

Выходными данными могут быть: обновляемая таблица в PostgreSQL, группа таблиц в ClickHouse, папка с данными в HDFS.

3. Программно-аппаратные требования

3.1. Аппаратные требования

Решение должно быть масштабируемым и работать в облачной среде.

Предполагается запуск на серверной инфраструктуре (не на мобильных устройствах).

Устройство для демонстрации – настольный ПК или ноутбук с доступом в интернет.

3.2. Программные требования (ожидаемый стек):

- Hadoop (HDFS) — обязательно, Hive — рекомендуемо
- ClickHouse
- PostgreSQL
- Kafka (для стриминга) — рекомендуемо
- Airflow (или аналог: Prefect, Dagster, NiFi), для оркестрации — обязательно
- Docker Compose – обязательно, Kubernetes (для оркестрации контейнеров) — рекомендуемо
- Языки: Python, SQL
- ML/ИИ: использование LLM для анализа данных, метаданных, генерации SQL/DDL и рекомендаций и генерации рекомендаций
- Фронтенд: веб-интерфейс (React, Vue и др.)
- Бэкенд: FastAPI, Django или аналог
- Контейнеризация: Docker

4. Требования к презентации/демонстрации

На финальной питч-сессии команда должна представить:

➔ Живую демонстрацию работы сервиса:
- Загрузка данных тестовых данных (CSV, JSON, XML);
- Анализ и рекомендация системы хранения;
- Создание пайплайна;
- Перенос данных в целевую систему (PostgreSQL, ClickHouse, HDFS);
- Автоматическое обновление по расписанию.

➔ Визуализацию архитектуры и логики принятия решений ИИ.

➔ Объяснение бизнес-ценности: кому поможет, как упростит работу, где может быть внедрено.

Презентация – до 7 минут, плюс 3 минуты на вопросы.

Формат: Презентация представляется в формате pptx или pdf + live demo (рекомендуется записать резервное видео на случай технических сбоев).

5. Требования к сопроводительной документации

К решению необходимо приложить:

1. README.md с описанием:
- Как запустить проект (локально и/или в облаке),
- Какие системы подключены,
- Как работает ИИ-модель,

- Скриншоты интерфейса.

2. Архитектурная схема решения (в формате .png или .drawio).
3. Краткий отчёт ИИ – пример вывода рекомендаций на тестовых данных.
4. Видео (до 3 минут) с демонстрацией ключевых функций (по желанию, но рекомендуется).

6. Ресурсы

6.1. Аппаратные ресурсы

Для развертывания различных систем в Docker Compose (AF, Clickhouse, PostgreSQL, etc):

Минимальные системные требования:

CPU: 8 ядра
RAM: 16 ГБ
Дисковое пространство: 20-30 ГБ (для образов и данных)
ОС: 64-битная (Linux/macOS/Windows 10+ с WSL2)

Минимальные системные требования по LLM:

1) квантованная русскоязычная модель 7B

RAM: 16 ГБ (минимум: 10 ГБ для 7B-модели + 6 ГБ для системных процессов)
GPU: Необязателен, но если есть: NVIDIA с 6 ГБ VRAM (GTX 1660 Super/RTX 3050)
CPU: 4-ядерный x86-64 с поддержкой AVX/AVX2 (Intel i5-6500+, Ryzen 3 2200G+)
SSD: 8 ГБ свободного места

2) рекомендуем модель с большим количеством параметров (но не более 32B), которая может быть запущена локально на 1 видеокарте NVIDIA A100/H100 80Gb)

6.2. Программные и вспомогательные ресурсы

Тестовые данные:

- *1 CSV-файл с обезличенными данными,*
- *1 JSON-файл с обезличенными данными,*
- *1 XML-файл с обезличенными данными*

7. Требования к сдаче решений

7.1. Промежуточная сдача

- Заявка на участие с кратким описанием подхода и с промежуточными результатами (не более 700 слов).
- Архитектурная схема (черновик).
- Подтверждение доступа к тестовым данным и системам.
- Демо-видео (1 минута) с первыми шагами (например: подключение к CSV, анализ структуры).

7.2. Финальная сдача

- Полный исходный код (GitHub/GitLab репозиторий).
- Рабочий веб-сервис (демо-версия в облаке или инструкция по локальному запуску).
- Все материалы из п. 5 (документация, презентация, видео).
- Ответы на вопросы.

8. Критерии оценки

8.1. Подход коллектива к решению задачи

- Чёткость плана, распределение ролей, креативность подхода.
- Учёт пользовательского опыта и реальных бизнес-сценариев.

8.2. Техническая проработка решения

- Корректность использования технологий (Airflow, Kafka, Kubernetes и др.).
- Надёжность, масштабируемость, автоматизация.
- Использование LLM для анализа данных и генерации рекомендаций (а не только предопределённые правила)

8.3. Соответствие решения поставленной задаче

- Реализация всех ключевых функций: анализ данных, рекомендации, ETL, UI.
- Поддержка разных источников и целей (включая стриминг через Kafka).

8.4. Эффективность решения в рамках поставленной задачи

- Скорость обработки, оптимизация запросов, эффективное хранение.
- Минимум ручного вмешательства.

8.5. Выступление коллектива на питч-сессии

- Ясность изложения, убедительность, качество презентации и демо.
- Ответы на вопросы жюри.

Дополнительно:

Лучшее решение может быть внедрено в работу Правительства Москвы.
Вы создаёте не просто прототип — вы создаёте будущее data engineering’а.