Тестовое задание: Создание системы для обработки жалоб клиентов с интеграцией публичных API

1. Интеграция с публичными API
Задача:
Разработать API для обработки жалоб клиентов с использованием публичных API.
Функционал:

1. Принимать POST-запросы с текстом жалобы (например, {"text": "Не приходит SMS-код"}).
2. Отправлять текст жалобы на внешний API для анализа тональности (например, **Sentiment Analysis by APILayer**).
3. Сохранять данные в SQLite с полями:

   id (уникальный идентификатор, автоинкремент).
   text (текст жалобы).
   status (open/closed, по умолчанию open).
   timestamp (время создания записи).
   sentiment (positive/negative/neutral).
   category (техническая/оплата/другое, по умолчанию другое).
4. Возвращать JSON-ответ с полями:

   id
   status
   sentiment
   category (если определена).

Технические требования:
Использовать **FastAPI** (Python) или **Express** (Node.js).
Реализовать обработку ошибок:
При недоступности внешнего API сохранять sentiment: "unknown".
Возвращать HTTP-статусы 500 для внутренних ошибок.
Исходный код разместить на **GitHub/GitLab** с инструкцией:
Установка зависимостей (requirements.txt/package.json).
Запуск приложения.
Примеры запросов через curl или Postman.
Примеры API для интеграции (обязательно):
Анализ тональности: Sentiment Analysis by APILayer (100 бесплатных запросов/месяц).
Опционально (доп. баллы):
Спам-фильтр: Spam Check by API Ninjas (50 запросов/день).
Геолокация по IP: IP API (без регистрации).

2. Определение категории жалобы с помощью ИИ
Задача:
Автоматически определять категорию жалобы (техническая, оплата, другое) с использованием:
OpenAI API (GPT-3.5 Turbo) или

Mistral-7B через Hugging Face (в Google Colab).

Требования:

Категория определяется сразу после сохранения жалобы в базу.

Для OpenAI: использовать промпт вида:

Определи категорию жалобы: "текст_жалобы". Варианты: техническая, оплата, другое. Ответ только одним словом.

Обновлять поле category в базе данных.

Если определение невозможно, сохранять category: "другое".

3. Автоматизация в n8n

Задача:

Настроить workflow в **n8n**, который:

1. Каждый час проверяет новые жалобы через API бэкенда (фильтр: status=open и timestamp за последний час).
2. Для жалоб категории техническая:

   Отправляет уведомление в Telegram-бота (создать через @BotFather).
   Меняет статус на closed.
3. Для жалоб категории оплата:

   Добавляет запись в Google Sheets (дата, текст жалобы, тональность).
   Меняет статус на closed.

Требования:

Использовать триггер schedule trigger и HTTP-запросы к API бэкенда.

Для Google Sheets: использовать сервисный аккаунт и OAuth2.

Предоставить скриншот рабочего workflow в n8n (включить узлы: HTTP-запрос, Telegram, Google Sheets, обновление статуса).

Дополнительные указания:

1. В README репозитория указать:
   - Настройку переменных окружения (API-ключи, токен бота).
   - Пример .env файла.

Часовой пояс и локация

• В каком часовом поясе вы находитесь?

• Готовы ли вы работать в часовом поясе, отличном от вашего?

• Есть ли у вас возможность синхронизироваться с командой в определенные часы (например, 2–4 часа в день)?

Опыт работы

• Какой у вас опыт в [укажите сферу, например, разработке, маркетинге, управлении проектами]?

• Какие задачи в предыдущих проектах/должностях были для вас наиболее сложными?

• Есть ли у вас портфолио или кейсы, которыми можно поделиться?

Доступность и время

• Сколько часов в неделю вы готовы уделять работе?

• Предпочитаете ли вы гибкий график или фиксированные рабочие часы?

• Как планируете совмещать эту работу с другими обязательствами (если есть)?