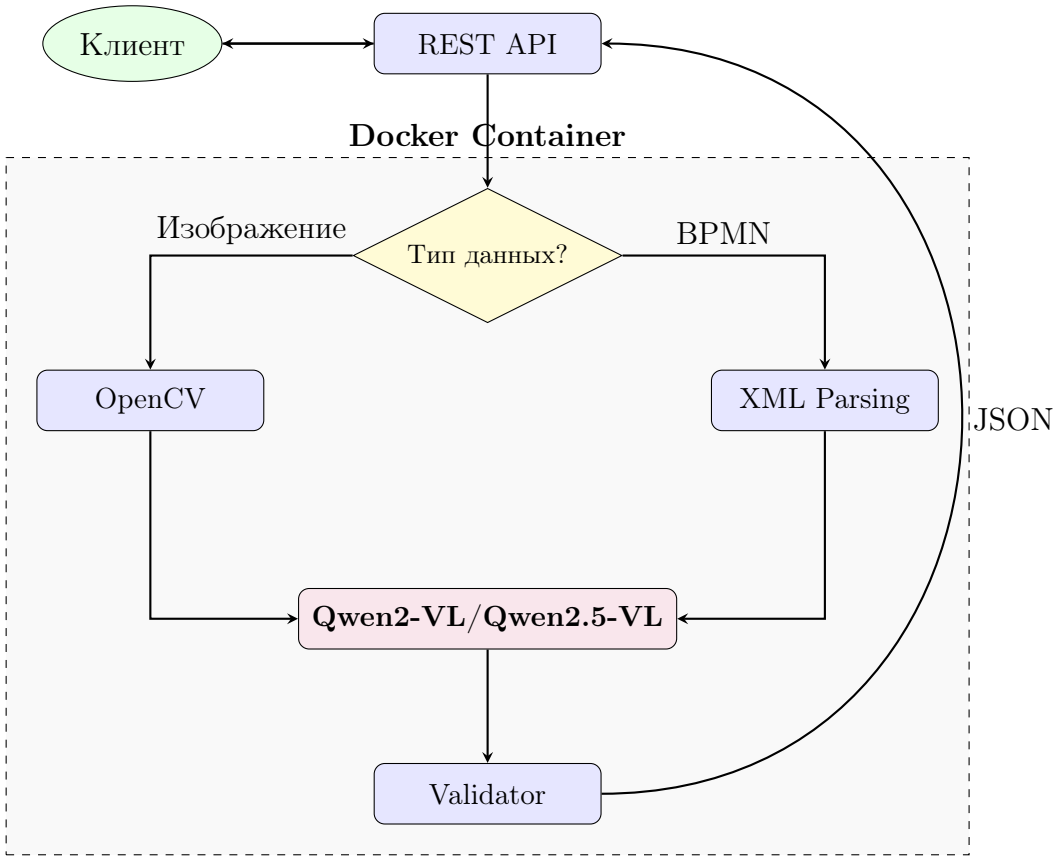
1 Описание архитектуры сервиса

Сервис реализует гибридную архитектуру обработки. В зависимости от типа входных данных (изображение/слайд или BPMN-диаграмма) система препроцессит данные, а после использует общую VLM для генерации финального алгоритма. Также в сервисе нет базы данных для хранения запросов - обработка происходит в реальном времени.

Общая схема сервиса:



2 Технологический стек

Стек оптимизирован и облегчен для ограниченных ресурсов (8GB VRAM), приоритетно используется GPU для ускорения инференса.

Категория	**Технология**	**Обоснование**
Язык	**Python 3.11**	Есть все необходимые библиотеки для работы с данными и моделями.
Backend	**FastAPI**	Асинхронная обработка. Удобная реализация REST API, поддержка загрузки файлов, автодокументация.
VLM	**Qwen2-VL/Qwen2.5-VL**	Отличные модели для обработки изображений и текста. Влезают в ресурсы.
Инференс	**Llama.cpp**	Эффективный запуск на CPU/GPU.
XML Parsing	**lxml/Regex**	Очистка BPMN от визуальных тегов.
UI	**Swagger UI**	Интерфейс для проверки запросов.
Инфраструктура	**Docker**	Единый контейнер.

3 План реализации (roadmap)

1. **Анализ данных и препроцессинг:**

   - Проверка используемых типов данных: BPMN XML, BPMN Image, схемы от руки, слайды. Написание скриптов нормализации изображений (коррекция размера, контрастность) и очистки XML.
   - В итоге должны получиться 4 типа данных для дальнейшей работы: BPMN XML, BPMN Image, схемы от руки, слайды.

2. **Интеграция VLM:**

   - Настройка Llama.cpp с моделью Qwen2.5-VL. Разработка промпта для извлечения алгоритмической структуры. Тестирование few-shot примеров.
   - В результате получаем рабочую модель для извлечения алгоритмической структуры из данных, которая запускается и отрабатывает все примеры.

3. **Разработка API:**

   - Создание эндпоинтов на FastAPI. Реализация прямого инференса и подключение Swagger UI.
   - По окончании получаем рабочие REST API эндпоинты.

4. **Упаковка в контейнер:**

   - Сборка Docker-образа. Тест производительности (уложиться в 20 сек).
   - В итоге сервис запускается в Docker-контейнере с помощью одной команды.

5. **Реализация обратной задачи:**

   - Реализация обратной задачи: из текста алгоритма генерировать диаграмму с использованием LLM моделей.
   - На выходе получаем модель для генерации диаграммы из текста алгоритма.