## 📘 ****Описание системы****

Нужно построить автоматизированную систему, которая принимает архивные документы по исполнительным производствам, извлекает из них смысловую информацию и передаёт результат в 1С в унифицированном формате.

### 💡 Что это за система?

Это интеллектуальный парсер судебных документов, который работает без участия человека и выполняет полный цикл:

1. Обходит заранее указанные папки (по справочнику кредиторов);
2. Извлекает текст из документов форматов PDF и DOCX;
3. Передаёт текст в нейросеть (AI) для понимания содержания;
4. Извлекает ключевые данные: номер ИП, дату, мероприятие, адресат, цель;
5. Проверяет корректность данных (валидация по шаблонам и справочникам);
6. Формирует итоговую структуру в формате JSON;
7. Отправляет результаты в 1С через API;
8. Фиксирует каждый этап: статус, ошибки, дубли, пропущенные данные.

### 🧠 Как построена логика?

Система модульная. Каждый компонент отвечает за свой этап:

* filewalker.py — находит и фильтрует документы для обработки;
* parser.py — извлекает текст и метаданные из файлов;
* validator.py — проверяет структуру и обязательные поля;
* ai\_client.py — обращается к AI через API, получает смысловые поля;
* exporter.py — формирует итоговый JSON;
* api\_client.py — передаёт результат в 1С по API;
* state\_manager.py — ведёт логи, статус и фиксацию ошибок.

Каждое действие логируется. Ошибки не блокируют общий процесс — документы с ошибками фиксируются отдельно.

### 🧩 Зачем это нужно?

В отделах исполнительного производства ежедневно обрабатываются сотни документов вручную — это медленно, неточно и не поддаётся аудиту. В частности:

* Документы могут быть утеряны или неправильно классифицированы;
* Сотрудники тратят время на рутинный перенос данных в 1С;
* Аналитика и контроль происходят "на глаз".

Система создаётся для:

* **Сокращения ручного труда**;
* **Ускорения загрузки в 1С**;
* **Снижения количества ошибок**;
* **Повышения прозрачности и аудируемости работы с документами**.

### 📊 Что должна давать система:

* 📁 Итоговый JSON-файл с валидными данными;
* ✅ Статус “OK” для успешно обработанных документов;
* ⚠️ Статус “ошибка” или “неполные данные” с причинами (например, нет номера ИП, AI не ответил);
* 📄 Полные логи всех операций: что, когда и почему произошло.

### 📈 Ключевые метрики качества:

| **Метрика** | **Описание** |
| --- | --- |
| % успешно обработанных | Сколько документов передано в 1С без ошибок |
| % ошибок | Документы, не прошедшие валидацию или не получившие ответ от AI |
| Время полного цикла | От старта обработки файла до ответа 1С |
| Уникальность имён файлов | Исключено затирание при одинаковых названиях |
| Стабильность | Любой сбой не прерывает работу всей цепочки |

### 🧭 Как разобраться в архитектуре:

* 🔗 **Полная архитектура и логика**: см. документ Архитектра системы парсера документов 1V2;
* 🔗 **Справочники и шаблоны** — в папке /configs/:
  + events.csv — мероприятия;
  + adresat.csv — адресаты;
  + validators.yaml — регулярки и шаблоны;
  + creditors\_to\_process.csv — список папок для обработки;
* 🔗 **Сценарий запуска**: раздел main.py + блок **10. Сквозной сценарий**;
* 🔗 **AI и prompt**: раздел **7.4.3** — логика извлечения цели и комментария;
* 🔗 **Формирование JSON для 1С**: см. раздел **8**;
* 🔗 **Ошибки и логирование**: см. раздел **9**.

## 🧾 Техническое задание (ТЗ)

### 📌 Название проекта:

**Автоматизированная система обработки судебных документов с передачей данных в 1С через API**

### 📄 Версия архитектуры:

v2 — см. документ **«Архитектра системы парсера документов 1V2»**

### 1️⃣ Цель проекта

Нужно разработать надёжную, модульную и отказоустойчивую систему, которая:

* Извлекает текст из документов форматов **PDF и DOCX**  
  → [см. раздел 7.2. Извлечение текста из файлов]
* Определяет ключевые поля через **AI и шаблоны**  
  → [см. раздел 7.4. Извлечение полей + 7.4.3 AI/шаблоны]
* Валидирует структуру и значения  
  → [см. раздел 7.5. Проверка полноты, дублей, статус]
* Формирует JSON-структуру для 1С  
  → [см. раздел 7.6. Итоговая структура на выходе]
* Передаёт данные в 1С по API  
  → [см. раздел 8. Загрузка в 1С]
* Ведёт полный лог обработки и ошибок  
  → [см. раздел 9. Логирование и аудит]

### 2️⃣ Объём работ специалиста

#### 🔍 2.1. Ознакомиться с архитектурой

* 📘 Основной документ: **«Архитектра системы парсера документов 1V2»**
* 📂 Структура папок и модулей: см. раздел **4. Структура проекта**
* 🔁 Сквозная логика: см. раздел **10. Запуск, координация и сквозной сценарий**

#### 🛠 2.2. Реализовать полный pipeline:

* main.py — централизованный запуск системы  
  → [см. 10.1 Общая логика запуска]
* filewalker.py — обход и фильтрация новых файлов  
  → [см. 7.1. Алгоритм обхода]
* parser.py — извлечение текста и предварительная структура  
  → [см. 7.2.3 Пример кода + 7.3.3]
* validator.py — валидация структуры и полей  
  → [см. 7.5.1–7.5.2]
* ai\_client.py — отправка текста в AI, получение результатов  
  → [см. 7.4.3 Выделение смысловых фрагментов + структура prompt]
* exporter.py — формирование итогового JSON  
  → [см. 7.6.1–7.6.2]
* api\_client.py — отправка в 1С  
  → [см. 8.2 Алгоритм автоматической загрузки]
* state\_manager.py — ведение логов, управление флагами  
  → [см. 9.1–9.2]

#### ⚙️ 2.3. Архитектурные требования:

* Модульность — каждый этап реализован независимо
* Отказоустойчивость — сбой одного шага не прерывает остальные
* Обязательное логирование по каждому этапу:
  + process\_log.json
  + error\_log.json
  + not\_processed.json
  + duplicates\_log.json

#### 🧾 2.4. Именование и контроль:

* Уникальные имена файлов (номер ИП + дата + имя)
* Повторная обработка не затирает предыдущие результаты
* Наличие pause.flag приостанавливает работу

#### ✅ 2.5. Условия передачи данных:

* Отправка в 1С только записей со статусом "OK"
* Проверка обязательных полей: номер ИП, дата, мероприятие, адресат
* Документы с неполными или ошибочными данными фиксируются отдельно

#### 🤖 2.6. Интеграция с AI:

* Используется заранее согласованный prompt → [см. 7.4.3]
* Ответ от AI — JSON с полями goal, comment
* Ошибки от AI не должны блокировать процесс

### 3️⃣ Результат работ

На выходе — система, которая:

* Автоматически обрабатывает документы;
* Генерирует валидный JSON для 1С;
* Сохраняет журналы и логи;
* Ведёт учёт ошибок, дублей, пропущенных данных;
* Поддерживает **тестовый и продуктивный режим** работы.

### 4️⃣ Контрольные точки (чек-лист)

| **Этап** | **Критерий готовности** |
| --- | --- |
| filewalker | Файлы отобраны, исключены дубли |
| parser | Текст извлечён, поля распознаны |
| validator | Все обязательные поля присутствуют и валидны |
| ai\_client | Ответ от AI получен, поля goal/comment извлечены |
| exporter | JSON выгрузка создана по структуре 1С |
| api\_client | Ответ от 1С получен, зафиксирован |
| logs | Все события и ошибки записаны в соответствующие журналы |

### 5️⃣ Формат обратной связи от специалиста

Специалист должен:

* Подтвердить понимание архитектуры;
* Уточнить неясные этапы обработки или связи между модулями;
* Протестировать систему на **тестовом наборе файлов**;
* Подтвердить, что каждый этап (от обхода до API 1С) работает как ожидается.