# ТЕХНИЧЕСКОЕ ЗАДАНИЕ НА СОЗДАНИЕ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ

Версия: 0.2 (черновик)

Дата: 10.08.2025

Автор: Боряков Никита Андреевич

Статус: Черновик

## 1. ВВЕДЕНИЕ И ЦЕЛИ ПРОЕКТА

## 1.1 Наименование разработки

Система автоматизации рутинных операций аналитика продаж с использованием AI-агента.

(далее — Система)

Краткое наименование:

Sales Analyst AI-Agent

## 1.2 Основание для разработки

Разработка осуществляется на основании:

- Внутренней инициативы аналитика отдела продаж;

- Необходимости автоматизации рутинных процессов:

- Создание и пере привязка торговых точек (ТТ) в ISA CRM;

- Сверка накладных из ISA в SFA;

- Новые процессы, которые аналитик должен производить на постоянной основе;

- Потребности в снижении трудозатрат, минимизации ошибок и ускорении обработки запросов от менеджеров;

- Большой текучки кадров и необходимости создать умную систему, которая будет справляться с рутинными задачами аналитика во время его отсутствия.

## 1.3 Источники разработки

|  |  |  |  |
| --- | --- | --- | --- |
| № | Документ / Источник | Формат | Автор |
| 1 | Product Backlog | Excel | Аналитик |
| 2 | Правила и рекомендации при создании ТТ в CRM | Excel | Аналитик |
| 3 | Справочники в системе 1С | Excel, 1С | Аналитик |
| 4 | Примеры запросов от менеджеров | Текстовые сообщения | Менеджеры |
| 5 | Техническая архитектура: operation-process-view | Структура в коде | Аналитик, ИИ-ассистент |
| 6 | Функциональные и не функциональные требования к системе, user story, tasks | Таблицы в excel | Аналитик |

## 1.4 Цель разработки

Целью разработки является создание масштабируемого, тестируемого и документированного программного обеспечения, предназначенного для автоматизации рутинных операций аналитика, а так же обучения этим вещам нового аналитика в случае увольнения старого.

Конечная цель — построение устойчивой, поддерживаемой и расширяемой системы, которая:

- Имеет чёткую архитектуру

- Поддерживает модульность и повторное использование кода

- Обеспечивает прозрачность процессов через логирование и уведомления

- Готова к интеграции с внешними системами (1С, CRM, DMS)

- Имеет корректную документацию и инструкции для обучения новых аналитиков

## 1.5 Основные задачи системы

|  |  |  |
| --- | --- | --- |
| № | Задача | Описание |
| 1 | Автоматизация создания ТТ в CRM | На основе данных из 1С или запроса менеджера |
| 2 | Перепривязка ТТ к новому ESR | По XCRM GUID, без затрагивания MRC-привязок |
| 3 | Сверка накладных с DMS | Проверка наличия накладных из Excel |
| 4 | Поддержка AI-агента | Чат-интерфейс для приёма запросов и запуска процессов |
| 5 | Фоновая обработка | Все процессы выполняются без блокировки GUI |
| 6 | Логирование и уведомления | Ошибки и успехи фиксируются в файл логов |
| 7 | Хранение данных | В PostgreSQL: ТТ, маппинг район → ESR, история операций |
| 8 | Экспорт результатов | В Excel с цветовой заливкой и пометками для быстрой проверки на успешность завершения процессов |

## 1.6 Ожидаемые результаты внедрения

|  |  |  |
| --- | --- | --- |
| № | Результат | Показатель |
| 1 | Сокращение времени на массовое создание ТТ | С 1–2 минут до 30 секунд |
| 2 | Уменьшение ошибок при вводе | 100% исключение ошибок |
| 3 | Повышение прозрачности | 100% логирование всех операций |
| 4 | Ускорение сверки накладных | С 6 часов до 15 минут |
| 5 | Готовность к масштабированию | Поддержка автономной работы через чат-бот |
| 6 | Лёгкость в обучении управления системой новым аналитиком | Чёткие инструкции и документация |

## 1.7 Обоснование необходимости разработки

В настоящее время все процессы выполняются вручную, что приводит к:

- Высокой трудоёмкости

- Ошибкам при вводе данных

- Задержкам в обработке запросов

- Сложности в обучении новых сотрудников

Автоматизация позволит:

- Освободить аналитика для более стратегических задач

- Обеспечить единообразие и качество данных

- Создать основу для дальнейшего развития (интеграция с 1С, AI-обучение)

## 1.8 Уровень документа

Настоящее ТЗ соответствует требованиям ГОСТ 34.602-89 и предназначено для внутреннего использования в рамках проекта.

## 2. ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

## 2.1 Общие положения

Система должна обеспечивать автоматизацию ключевых процессов аналитика:

- Создание торговой точки (ТТ) в CRM по данным из 1С или текстового запроса

- Пере привязка ТТ к новому ESR по XCRM GUID

- Сверка накладных с системой DMS на основе Excel-файла

- Поддержка AI-агента, способного принимать запросы в чате, уточнять данные и запускать процессы

Система поддерживает два режима ввода:

- Ручной ввод через графический интерфейс

- AI-чат: текстовый запрос → извлечение данных → уточнение → запуск процесса

Все процессы выполняются в фоне, без блокировки GUI.

## 2.2 Функциональные требования

Функциональные требования описаны в таблице «Функциональные требования к системе.xlsx» и приложены к настоящему ТЗ.

## 2.3 Бизнес-правила

Бизнес-правила при создании системы описаны в таблице «Бизнес-правила.xlsx» и приложены к ТЗ.

## 2.4 Потоки данных

## 2.4.1 Поток: Создание новой ТТ

```mermaid

graph TD

A[Менеджер отправляет запрос] > B{Есть XCRM GUID?}

B Нет > C[AI извлекает: owner, address, region\_code]

C > D[Поиск esr\_code по region\_code в PostgreSQL]

D > E[Модель предсказывает: point\_type, area, assortment]

E > F[Формирование TradePoint]

F > G[Валидация обязательных полей]

G > H[Запуск процесса в фоне]

H > I[Selenium: заполнение формы в CRM]

I > J[Первичное сохранение → генерация XCRM GUID]

J > K[Привязка к ESR: ввод кода → выбор направления]

K > L[Финальное сохранение]

L > M[Экспорт отчёта в Excel]

M > N[Уведомление: успех]

```

## 2.4.2 Поток: Перепривязка ТТ

```mermaid

graph TD

A[Менеджер отправляет запрос с XCRM GUID] > B[AI распознаёт: перепривязка]

B > C[Selenium: поиск ТТ по XCRM GUID]

C > D[Открытие формы редактирования]

D > E[Анализ таблицы привязок]

E > F{Строка содержит "ESR"?}

F Да > G[Удалить (деактивировать)]

F Нет > H[Сохранить]

G > I[Добавление новой привязки: ESR + направление]

I > J[Сохранение изменений]

J > K[Экспорт отчёта]

K > L[Уведомление: успех]

```

## 2.4.3 Поток: Сверка накладных

```mermaid

graph TD

A[Загрузка Excel-файла] > B[Обновление сводных таблиц (COM/xlwings)]

B > C[Фильтрация: SFA=пусто, ISA≠0]

C > D[Копирование на отдельный лист]

D > E[Запуск сверки в фоне]

E > F{Накладная есть в DMS?}

F Да > G[Оставить без изменений]

F Нет > H[Залить красным цветом]

H > I[Добавить пометку: "нет в DMS"]

G > I

I > J[Сохранение результата в Excel]

J > K[Уведомление: завершено]

```

## 2.5 Структура сущности `TradePoint`

```python

@dataclass

class TradePoint:

owner: str #Контрагент (владелец ТТ)

address: str #Адрес ТТ

region\_code: str #Код района маршрута (например, Н02)

# Предсказываемые поля (моделью)

point\_type: str

# Поля проставляемые в зависимости от point\_type, названия ТТ и контрагента

assortment: str

chain\_name: str

esr\_code: str

# Заполняемые системой

location\_type: str

area: str

xcrm\_guid: Optional[str] = None

status: str = "pending"

error\_message: Optional[str] = None

@property

def name(self) -> str:

return f"{self.owner} {self.address}"

def to\_json(self) -> dict: ...

```

## 2.6 Триггеры и реакции системы

|  |  |
| --- | --- |
| Триггер | Реакция системы |
| Запрос в чате: "Создать ТТ: ООО Ромашка, Новосибирск, ул. Ленина 15, район Н02" | AI извлекает данные, проверяет наличие region\_code → esr\_code, запрашивает уточнения (если нужно), формирует JSON |
| Пользователь нажимает кнопку "Запустить" на вкладке "Создание ТТ" | Проверка обязательных полей → запуск процесса в фоне → Selenium заполняет CRM |
| Загружен Excel с накладными | Обновление данных → фильтрация → копирование → запуск сверки |
| Сервер DMS недоступен | Ошибка логируется, процесс продолжается для следующей накладной |
| Процесс завершён (успех/ошибка) | Показ уведомления, экспорт отчёта, обновление статуса |

## 3. НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

## 3.1 Общие положения

Система должна обеспечивать стабильную, безопасную и удобную работу в условиях локальной эксплуатации аналитиком.

Нефункциональные требования охватывают:

- Надёжность и отказоустойчивость

- Производительность и UX

- Безопасность конфиденциальных данных

- Требования к аппаратному и программному обеспечению

- Ограничения по использованию

## 3.2 Требования к надёжности

| ID | Требование | Источник | Категория |

||-|-|-|

| NFR-REL-01 | Система должна продолжать работу при временной недоступности DMS (например, сервер упал, таймаут) | T1.14, US5.4 | Обработка ошибок |

| NFR-REL-02 | При недоступности DMS система должна логировать ошибку и переходить к следующей накладной, не прерывая процесс сверки | T1.14, US5.4 | Обработка ошибок |

| NFR-REL-03 | Система должна обрабатывать капчу, редиректы и смену сессии в CRM/DMS, повторяя попытку входа при необходимости | US4.4, T1.2 | Стабильность Selenium |

| NFR-REL-04 | При падении процесса в фоне (QThread) система должна корректно завершить поток и зафиксировать ошибку в логах | T1.4 | Безопасность |

| NFR-REL-05 | Система должна поддерживать повторные попытки (retry) при сетевых ошибках (например, 502, 503) с экспоненциальной задержкой | T1.14 | Надёжность |

| NFR-REL-06 | Все операции и ошибки должны логироваться в файл `logs/app.log` с уровнем DEBUG | US1.5, T1.6 | Логирование |

| NFR-REL-07 | Логи должны содержать: дату, тип операции, входные данные, статус, сообщение об ошибке, XCRM GUID (если есть) | US1.5 | Логирование |

| NFR-REL-08 | Перед запуском процесса система должна проверять наличие обязательных полей и не начинать операцию при их отсутствии | US1.11, T1.10 | Защита от ошибок |

| NFR-REL-09 | При перепривязке ESR система не должна удалять привязки к MRC, даже если они находятся в одной таблице | US3.2, T1.2 | Целостность данных |

| NFR-REL-10 | После сбоя система должна продолжать обработку следующих элементов списка (накладных, ТТ) | T1.14 | Восстановление |

## 3.3 Требования к производительности и UX

| ID | Требование | Источник | Категория |

||-|-|-|

| NFR-PERF-01 | GUI не должен блокироваться во время выполнения процессов (реализовать через QThread) | US1.3, US2.7 | Производительность |

| NFR-PERF-02 | Время отклика уведомления — не более 2 секунд после завершения процесса | FR-11 | UX |

| NFR-PERF-03 | Поддержка обработки до 1000 строк в Excel без зависаний | US2.1 | Производительность |

| NFR-PERF-04 | Интерфейс должен содержать иконки на кнопках без текста | US5.1 | UX |

| NFR-PERF-05 | Выходной Excel-файл должен автоматически открываться после завершения процесса | FR-19 | UX |

| NFR-PERF-06 | Валидация данных должна выполняться до запуска процесса (Pydantic) | T1.10 | Надёжность |

| NFR-PERF-07 | Система должна быть устойчива к таймаутам и капчам в CRM/DMS | US4.4 | Стабильность |

## 3.4 Требования к безопасности

| ID | Требование | Источник | Категория |

||-|-|-|

| NFR-SEC-01 | Конфиденциальные данные (логины, пароли, API-ключи) не должны сохраняться в репозитории или внешних файлах | US6.3 | Защита данных |

| NFR-SEC-02 | Пароли и API-ключи вводятся пользователем в GUI и хранятся локально на устройстве | US6.3 | Защита данных |

| NFR-SEC-03 | В GUI поля с паролями должны отображаться с маской (•••••) | — | UX / Безопасность |

| NFR-SEC-04 | Все запросы к OpenRouter API должны выполняться по HTTPS | — | Защита данных |

| NFR-SEC-05 | Система не должна логировать пароли, API-ключи или ИНН | US1.5 | Защита данных |

| NFR-SEC-06 | При первом запуске пользователь обязан заполнить настройки (CRM/DMS, OpenRouter) для активации функциональности | — | Контроль доступа |

| NFR-SEC-07 | Пути к файлам (`C:\Data\...`) должны быть защищены правами ОС (только пользователь и администратор) | — | Защита файлов |

> ✅ Примечание:

> Система рассчитана на одного пользователя. Аутентификация на уровне ОС.

> Шифрование `settings.json` не требуется на этапе MVP.

## 3.5 Требования к техническому обеспечению

## 3.5.1 Аппаратное обеспечение

| Параметр | Минимальные требования | Рекомендуемые требования |

||||

| Процессор | Intel Core i3 или аналог | Intel Core i5 / i7 или выше |

| Оперативная память | 8 ГБ | 16 ГБ |

| Свободное место на диске | 5 ГБ | 10 ГБ |

| Видеокарта | Интегрированная | Интегрированная |

| Монитор | Разрешение 1366×768 | Разрешение 1920×1080 |

## 3.5.2 Программное обеспечение

| Компонент | Требование |

||-|

| Операционная система | Windows 10, Windows 11 (64-bit) |

| Интерпретатор Python | Python 3.10 – 3.12 |

| Библиотеки Python | PyQt5, selenium, pandas, openpyxl, xlwings, psycopg2, pydantic |

| Браузер | Google Chrome (актуальная версия) |

| Microsoft Excel | Excel 2016 или новее (для COM-доступа) |

| PostgreSQL | PostgreSQL 14+ (локально или в сети) |

| Дополнительно | ChromeDriver (совместимый с версией Chrome) |

## 3.5.3 Сетевые требования

| Параметр | Требование |

||-|

| Подключение к интернету | Обязательно (для доступа к CRM и DMS) |

| Скорость соединения | Минимум 10 Мбит/с (рекомендуется 50 Мбит/с) |

| Доступ к CRM | Через корпоративную сеть (внутренний IP или домен) |

| Доступ к DMS | Через корпоративную сеть (внутренний IP или домен) |

| Доступ к PostgreSQL | Локально или через внутреннюю сеть |

| Доступ к файлам | Локальные или сетевые пути (UNC) |

## 3.6 Ограничения по использованию

| Ограничение | Описание |

|-|-|

| Многопользовательский режим | Не поддерживается. Система рассчитана на одного пользователя на рабочей станции |

| Кроссплатформенность | Не поддерживается. Только Windows (из-за Excel + COM) |

| Автономная работа | Возможна, но с ограничениями: без доступа к CRM/DMS/Excel — только просмотр логов и истории |

| Масштабирование | На текущем этапе — нет. В будущем — через интеграцию с 1С и чат-бот |

| Резервное копирование | Не реализовано. Данные в PostgreSQL и Excel считаются временными до интеграции с 1С |

## 4. ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ

## 4.1 Общие положения

Система использует структурированные и полуструктурированные данные, поступающие из внешних и внутренних источников.

Основные цели информационного обеспечения:

- Обеспечение единого источника правды для справочников (например, `region\_code → esr\_code`)

- Поддержка быстрого поиска и фильтрации данных о торговых точках

- Хранение истории операций для анализа и отладки

- Временное хранение входных и выходных файлов в формате Excel

Информационное обеспечение включает:

- Базу данных PostgreSQL

- Локальные Excel-файлы

- Внутренние структуры данных (например, `TradePoint`)

## 4.2 Источники данных

| Источник | Тип | Формат | Назначение | Частота обновления |

||-||-|-|

| 1С (будущее) | Внешняя система | API / SQL | Получение данных о ТТ, контрагентах, маршрутах | Ежедневно |

| CRM | Веб-интерфейс | HTML | Создание ТТ, привязка ESR | В реальном времени |

| DMS | Веб-интерфейс | HTML | Проверка наличия накладных | В реальном времени |

| Excel (входной) | Локальный файл | `.xlsx` | Список накладных для сверки | При каждом запуске |

| Excel (маппинг) | Локальный файл | `.xlsx` | Маппинг `region\_code → esr\_code` | При старте системы |

| PostgreSQL | Локальная / сетевая БД | Реляционная | Хранение данных о ТТ, маппинге, логах | Постоянно |

| OpenRouter API | Внешний API | JSON | Парсинг запросов, генерация JSON | При каждом запросе в чате |

> ✅ Примечание:

> На этапе MVP 1С не интегрирована. Все данные из 1С извлекаются вручную и вводятся в систему.

## 4.3 Структура базы данных

База данных реализована в PostgreSQL 14+ и содержит следующие таблицы:

## 4.3.1 Таблица `trade\_points`

| Поле | Тип | Описание | Обязательность |

||-|-||

| `id` | SERIAL | Уникальный идентификатор | Да |

| `owner` | VARCHAR(255) | Контрагент (владелец ТТ) | Да |

| `address` | TEXT | Адрес торговой точки | Да |

| `region\_code` | VARCHAR(10) | Код района маршрута (например, Н02) | Да |

| `inn` | VARCHAR(12) | ИНН контрагента | Нет |

| `point\_type` | VARCHAR(50) | Вид ТТ (например, "Counter") | Нет |

| `assortment` | VARCHAR(100) | Тип ассортимента (например, "Весь ассортимент") | Нет |

| `location\_type` | VARCHAR(50) | Местоположение (например, "Street") | Нет |

| `area` | VARCHAR(20) | Площадь (например, "1-50") | Нет |

| `chain\_name` | VARCHAR(100) | Название сети (например, "NO CHAIN") | Нет |

| `esr\_code` | VARCHAR(20) | Код ESR для привязки | Да (до привязки) |

| `xcrm\_guid` | UUID | Уникальный идентификатор ТТ в CRM | После создания |

| `status` | VARCHAR(20) | Статус: pending, created, failed, reattached | Да |

| `error\_message` | TEXT | Описание ошибки (если есть) | Нет |

| `created\_at` | TIMESTAMP | Дата и время создания | Да |

| `updated\_at` | TIMESTAMP | Дата и время последнего изменения | Да |

> ✅ Индексы:

> - `idx\_xcrm\_guid` — по `xcrm\_guid`

> - `idx\_region\_code` — по `region\_code`

> - `idx\_status` — по `status`

## 4.3.2 Таблица `region\_esr\_mapping`

| Поле | Тип | Описание | Обязательность |

||-|-||

| `id` | SERIAL | Уникальный идентификатор | Да |

| `region\_code` | VARCHAR(10) | Код района (например, Н02) | Да |

| `esr\_code` | VARCHAR(20) | Соответствующий ESR-код | Да |

| `direction` | VARCHAR(100) | Направление деятельности (например, "ООО Континент 'Абакан'") | Да |

| `created\_at` | TIMESTAMP | Дата добавления | Да |

| `updated\_at` | TIMESTAMP | Дата обновления | Да |

> ✅ Уникальность:

> - Ограничение `UNIQUE(region\_code)` — один район → один ESR

> - Синхронизация с Excel при старте системы

## 4.3.3 Таблица `invoice\_reconciliation`

| Поле | Тип | Описание | Обязательность |

||-|-||

| `id` | SERIAL | Уникальный идентификатор | Да |

| `invoice\_number` | VARCHAR(50) | Номер накладной | Да |

| `isa\_value` | DECIMAL | Значение ISA | Да |

| `sfa\_value` | TEXT | Значение SFA (пусто / число) | Да |

| `in\_dms` | BOOLEAN | Есть ли накладная в DMS | Да |

| `check\_date` | TIMESTAMP | Дата проверки | Да |

| `error\_message` | TEXT | Описание ошибки (если DMS недоступен) | Нет |

> ✅ Индексы:

> - `idx\_invoice\_number` — для быстрого поиска

> - `idx\_check\_date` — для анализа по периодам

## 4.3.4 Таблица `contractors`

| Поле | Тип | Описание | Обязательность |

||-|-||

| `id` | SERIAL | Уникальный идентификатор | Да |

| `inn` | VARCHAR(12) | ИНН контрагента | Да |

| `name` | VARCHAR(255) | Название контрагента | Да |

| `current\_chain` | VARCHAR(100) | Текущее название сети (например, "Wholesale") | Да |

| `last\_updated` | TIMESTAMP | Дата последнего обновления | Да |

| `is\_active` | BOOLEAN | Активен ли контрагент | Да |

> ✅ Цель:

> Поддержка сценария, когда контрагент меняет формат сети (например, с NO CHAIN на Wholesale), что требует пересоздания ТТ в CRM.

## 4.4 Форматы входных и выходных данных

## 4.4.1 Входные данные

| Источник | Формат | Структура |

||||

| GUI (вручную) | Форма | `owner`, `address`, `region\_code`, `inn` (опционально) |

| AI-чат | Текстовый запрос | "Создать ТТ: ООО Ромашка, Новосибирск, ул. Ленина, 15, район Н02" |

| Excel (накладные) | `.xlsx` | Сводная таблица с колонками: `Номер накладной`, `ISA`, `SFA` |

| Excel (маппинг) | `.xlsx` | Таблица: `Район маршрута`, `ESR-код`, `Направление деятельности` |

## 4.4.2 Выходные данные

| Назначение | Формат | Структура |

||||

| Отчёт о создании ТТ | `.xlsx` | Колонки: `XCRM GUID`, `Статус`, `Ошибка (если есть)` |

| Отчёт о сверке накладных | `.xlsx` | Исходный файл с цветовой заливкой: красный — нет в DMS |

| Логи | `.log` | Строки: `timestamp | level | operation | status | message` |

| Внутренний JSON (для процессов) | `dict` | `{ "owner": "...", "address": "...", "region\_code": "Н02", ... }` |

> ✅ Автоматическое открытие:

> После завершения процесса выходной Excel-файл открывается автоматически через `os.startfile()`.

## 4.5 Правила обработки данных

| Правило | Описание |

||-|

| BR-DATA-1 | При создании ТТ поле `name` формируется как `{owner} {address}` |

| BR-DATA-2 | При отсутствии `esr\_code` система ищет его в `region\_esr\_mapping` по `region\_code` |

| BR-DATA-3 | Если `region\_code` не найден в БД, AI уточняет у пользователя |

| BR-DATA-4 | При ручном указании `esr\_code` пара `region\_code → esr\_code` сохраняется в БД |

| BR-DATA-5 | При перепривязке удаляются только привязки, содержащие "ESR" в названии |

| BR-DATA-6 | Если `point\_type`, `assortment`, `location\_type`, `area` не указаны — система использует модель классификации для предсказания на основе истории и контекста |

| BR-DATA-7 | Дефолтные значения (`Counter`, `1-50`, `Street`) используются только если модель не может сделать предсказание |

| BR-DATA-8 | Процесс не запускается при отсутствии: `owner`, `address`, `region\_code` |

| BR-DATA-9 | Все операции логируются в `logs/app.log` и в `trade\_points.error\_message` |

## 4.6 Схема синхронизации данных

```mermaid

graph TD

A[Excel: region\_esr\_mapping.xlsx] >|При старте| B(PostgreSQL: region\_esr\_mapping)

B > C[AI-Agent: поиск ESR по region\_code]

C > D[Process: создание ТТ / сверка]

D > E[CRM / DMS]

D > F[PostgreSQL: trade\_points, invoice\_reconciliation]

F > G[Excel: отчёт (выходной)]

G > H[Автоматическое открытие]

```

> ✅ Ключевые принципы:

> - PostgreSQL — единый источник правды для справочников

> - Excel — временный источник и приёмник

> - Синхронизация маппинга — при запуске приложения

## 5. АРХИТЕКТУРА И ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

## 5.1 Общие положения

Система предназначена для локального использования аналитиком и не требует серверной инфраструктуры.

Все конфигурационные данные хранятся локально в приложении, доступны через вкладку "Настройки" в GUI.

Никакие конфиденциальные данные (логины, пароли, API-ключи) не сохраняются в репозитории или внешних файлах.

Раздел объединяет положения из старых разделов:

- "Техническое обеспечение"

- "Инструменты разработки и сопровождения"

- "Управление настройками через GUI"

## 5.2 Программные компоненты системы

| Компонент | Назначение | Версия / Требование |

||-|-|

| Python | Основной язык программирования | 3.10 – 3.12 |

| PyQt5 | Графический интерфейс (GUI) | >= 5.15 |

| Selenium | Автоматизация браузера (CRM, DMS) | >= 4.0 |

| pandas | Обработка данных Excel | >= 1.3 |

| openpyxl | Чтение/запись Excel (без COM) | >= 3.0 |

| xlwings | Работа с Excel через COM (обновление данных) | >= 0.27 |

| psycopg2 | Подключение к PostgreSQL | >= 2.9 |

| pydantic | Валидация данных, JSON | >= 1.10 |

> ✅ Примечание:

> Все зависимости указаны в файле `requirements.txt`, который не включается в ТЗ, но является частью релиза.

## 5.3 Внешние системы и точки интеграции

| Система | Тип доступа | Протокол | Назначение |

||-|-|-|

| CRM | Веб-интерфейс | HTTPS | Создание ТТ, привязка ESR |

| DMS | Веб-интерфейс | HTTPS | Проверка наличия накладных |

| 1С | Файловый обмен (будущее) | SMB / API | Получение данных о ТТ |

| Microsoft Excel | Локальный файл | COM / openpyxl | Загрузка и обработка данных |

| PostgreSQL | Локальная / сетевая БД | TCP/IP | Хранение данных о ТТ и маппинге |

| OpenRouter API | API | HTTPS | Парсинг запросов от пользователя, генерация JSON для запуска процессов |

> ✅ Обновление от вашего описания:

> - OpenRouter API используется как единая точка доступа к LLM (Qwen, GPT, Claude и др.)

> - Пользователь может выбрать модель, изменить API-ключ, обновить URL API

> - При первом запуске — обязательное заполнение настроек для активации функциональности

## 5.4 Модель хранения настроек

Настройки системы хранятся локально на устройстве пользователя в виде сериализованной структуры (например, `settings.json` или pickle), не выгружается в репозиторий.

Поля настроек:

| Группа | Параметр | Описание |

|-|-|-|

| Авторизация | CRM: логин, пароль | Вводится пользователем |

| | DMS: логин, пароль | Вводится пользователем |

| API Интеграции | OpenRouter: API URL | По умолчанию: `https://openrouter.ai/api/v1` |

| | OpenRouter: API Key | Вводится пользователем |

| | OpenRouter: Модель | Выбор из списка (Qwen, GPT, Claude и др.) |

| Системные ссылки | CRM URL | Жёстко задан в коде, может быть обновлён в hotfix |

| | DMS URL | Жёстко задан в коде, может быть обновлён в hotfix |

| Файловые пути | Входной Excel | Фиксированный путь: `C:\Data\input.xlsx` |

| | Выходной отчёт | Фиксированный путь: `C:\Data\output.xlsx` |

| | 1С сервер | Адрес сервера 1С (будущее) |

> ✅ Поведение при первом запуске:

> - Пользователь перенаправляется на вкладку "Настройки"

> - Не может использовать функции, зависящие от конфигурации, пока не заполнит:

> - CRM/DMS логин/пароль

> - OpenRouter API Key

> - В интерфейсе есть кнопка "Получить API Key" → открывает браузер с `https://openrouter.ai/keys`

## 5.5 Управление настройками через GUI

| Элемент | Описание |

||-|

| Вкладка "Настройки" | Централизованное управление всеми параметрами |

| Поле ввода API Key | С маской скрытия пароля |

| Кнопка "Проверить подключение" | Для CRM, DMS, OpenRouter |

| Кнопка "Открыть OpenRouter" | Переход на сайт для получения ключа |

| Сохранение | Локально, без шифрования (в защищённой среде пользователя) |

## 5.6 Обновление системных URL (hotfix)

| Система | Механизм обновления |

||-|

| CRM URL | Жёстко задан в коде → обновляется в `hotfix`-ветке при изменении |

| DMS URL | То же |

| OpenRouter API URL | Может быть изменён пользователем в настройках |

> ✅ Процедура:

> При поступлении жалобы на недоступность системы → разработчик вносит правку в URL → собирает новую версию → выпускает в тот же день.

## 5.7 Пути к файлам

| Тип файла | Путь | Примечание |

|||-|

| Входной Excel | `C:\Data\input.xlsx` | Автоматически загружается при старте процесса |

| Выходной отчёт | `C:\Data\output.xlsx` | Открывается автоматически после завершения процесса |

| Логи | `logs/app.log` | Создаётся при первом запуске |

> ✅ Автоматическое открытие:

> После завершения любого процесса (создание ТТ, сверка) — выходной файл открывается автоматически через `os.startfile()`.

## 5.8 Архитектура интеграции

```mermaid

graph LR

A[Пользователь] > B[GUI (PyQt5)]

B > C[AI-Agent (OpenRouter)]

C > D[Data Processing]

D > E[Operation: create\_tt / reconcile]

E > F[Selenium → CRM]

E > G[Selenium → DMS]

D > H[PostgreSQL]

B > I[Excel (openpyxl / xlwings)]

H > J[Кэш для AI]

I > J

J > C

```

> ✅ Ключевые принципы:

> - Все внешние вызовы — через модули `infrastructure`

> - Данные — через PostgreSQL как единый источник правды

> - GUI — не зависит напрямую от Selenium или Excel

## 6. ЖИЗНЕННЫЙ ЦИКЛ РАЗРАБОТКИ

## 6.1 Общие положения

Разработка системы осуществляется по итеративной модели (Agile/Scrum) с длительностью спринта 2 недели.

Цель — построение MVP (Minimum Viable Product) в течение первого спринта, способного выполнять ключевые функции:

- Создание ТТ в CRM по запросу из GUI или чата

- Перепривязка ТТ к новому ESR

- Сверка накладных с DMS

- Формирование отчётов в Excel

Последующие спринты направлены на стабилизацию, расширение и подготовку к интеграции с 1С.

## 6.2 Стадии разработки

| Стадия | Цель | Продолжительность | Результат |

|||-||

| 1. Формирование ТЗ | Сбор и формализация требований, утверждение v0.2 | 1 неделя | Утверждённое ТЗ |

| 2. Реализация (спринты) | Поэтапная разработка функциональности | 6 недель (3 спринта) | Работающая система (MVP + расширения) |

| 3. Тестирование | Проверка функциональности, надёжности, безопасности | Параллельно со спринтами | Отчёт о приёмо-сдаточных испытаниях |

| 4. Внедрение | Установка, настройка, обучение пользователя | 1 неделя | Система в эксплуатации |

| 5. Сопровождение | Поддержка, исправление ошибок, доработки | Постоянно | Обновления, hotfix-релизы |

## 6.3 План спринтов

📆 Спринт 1: MVP (Minimum Viable Product)

Цель: Создать минимально жизнеспособную версию системы.

| ID | Тип | Задача | Приоритет | Описание |

||-||-|-|

| T1.1 | Task | Реализовать dataclass `TradePoint` | Высокий | Поля: owner, address, region\_code, esr\_code, point\_type и др. |

| T1.2 | Task | Реализовать POM для CRM | Высокий | Страницы: вход, создание, редактирование ТТ |

| T1.3 | Task | Реализовать POM для DMS | Высокий | Страницы: вход, выбор подсистемы, поиск накладной |

| T1.4 | Task | Настроить фоновый поток (QThread) | Высокий | Без блокировки GUI |

| T1.6 | Task | Реализовать логирование | Высокий | В `logs/app.log`, уровень DEBUG |

| T1.11 | Task | Реализовать GUI с вкладками | Высокий | Вкладки: "Создание ТТ", "Сверка", "AI-чат" |

| US1.2 | User Story | Автоматизация создания ТТ через Selenium | Высокий | Заполнение формы в CRM |

| US2.1–US2.7 | User Story | Реализовать сверку накладных | Высокий | Загрузка → фильтрация → проверка → заливка |

| US1.4 | User Story | Реализовать уведомления | Высокий | Всплывающее окно: успех/ошибка |

| US1.13 | User Story | Реализовать экспорт отчёта | Высокий | Excel с GUID, статусом, ошибкой |

> ✅ Критерии приёма MVP:

> - Можно создать ТТ через GUI

> - Можно сверить накладные из Excel

> - Процессы работают в фоне

> - Есть уведомления и логи

> - Приложение не падает при ошибках DMS

📆 Спринт 2: AI-агент и интеллектуальная обработка

Цель: Внедрить AI-агента, модель классификации, хранение данных в PostgreSQL.

| ID | Тип | Задача | Приоритет | Описание |

||-||-|-|

| T1.7 | Task | Настроить PostgreSQL | Средний | Таблицы: `trade\_points`, `region\_esr\_mapping`, `invoice\_reconciliation` |

| T1.8 | Task | Реализовать синхронизацию маппинга | Средний | Обновление `region\_code → esr\_code` между Excel и БД |

| T1.12 | Task | Интегрировать LLM (OpenRouter) | Средний | Парсинг запросов, генерация JSON |

| US3.1–US3.3 | User Story | Реализовать перепривязку ESR | Высокий | По XCRM GUID → удалить ESR → добавить новый |

| US1.6 | User Story | Реализовать предсказание атрибутов | Средний | Модель предсказывает: `point\_type`, `area`, `location\_type` |

| US1.8 | User Story | Сохранять новые ESR в БД | Средний | При ручном вводе — сохранять в `region\_esr\_mapping` |

| US1.9–US1.10 | User Story | Реализовать AI-чат | Высокий | Диалог: запрос → уточнение → JSON → запуск |

> ✅ Критерии приёма:

> - Можно писать в чат: "Создать ТТ: ООО Ромашка, Новосибирск, ул. Ленина, 15, район Н02"

> - AI формирует JSON и запускает процесс

> - При отсутствии ESR — AI уточняет

> - Новые пары `region\_code → esr\_code` сохраняются в БД

> - Можно перепривязать ТТ к новому ESR

📆 Спринт 3: Стабилизация и подготовка к 1С

Цель: Устранить технический долг, улучшить UX, подготовить систему к будущей интеграции.

| ID | Тип | Задача | Приоритет | Описание |

||-||-|-|

| T1.14 | Task | Обработка ошибок DMS | Высокий | Логирование, продолжение при падении сервера |

| T1.10 | Task | Валидация JSON | Средний | Проверка обязательных полей перед запуском |

| US4.2 | User Story | Экспорт из PostgreSQL в Excel | Средний | Данные с фильтрами и срезами |

| — | Task | Добавить таблицу `contractors` | Средний | Хранение: ИНН, название, текущая сеть |

| — | User Story | Реализовать обогащение `chain\_name` | Средний | Использовать `current\_chain` при создании ТТ |

| — | Task | Подготовить заглушку для 1С | Низкий | Модуль `integration/1c.py` (заготовка) |

| — | Task | Обновить документацию | Средний | Руководства, описание программы, ПМИ |

> ✅ Критерии приёма:

> - Система не падает при сетевых ошибках

> - Все процессы валидируются

> - Есть возможность анализировать данные из БД в Excel

> - При смене сети у контрагента — можно пересоздать ТТ

> - Документация актуальна

## 6.4 Ответственные роли

| Роль | Ответственность |

|||

| Аналитик (заказчик) | Формулировка требований, приёмка спринтов, тестирование, наполнение данными |

| Разработчик | Реализация модулей, POM, GUI, интеграций, тестирование |

| QA / Аналитик | Проверка соответствия ТЗ, составление ПМИ, фиксация багов |

> ✅ Примечание:

> На текущем этапе аналитик и разработчик — одно лицо, но роли разделены для соблюдения процесса.

## 6.5 Критерии приёма системы

| Критерий | Описание |

||-|

| ✅ Функциональность | Все user story из спринтов 1–3 реализованы и протестированы |

| ✅ Надёжность | Система не падает при ошибках DMS, логирует все операции |

| ✅ Производительность | Процессы выполняются в фоне, GUI не блокируется |

| ✅ Безопасность | Конфиденциальные данные не логируются, хранятся локально |

| ✅ Документация | Все документы (ТЗ, руководства, ПМИ) утверждены |

| ✅ Обучение | Аналитик может самостоятельно использовать все функции |

## 6.6 Порядок контроля и приёмки

| Этап | Действие |

||-|

| Еженедельно | Статус-митинг: обсуждение прогресса, блокеров |

| Конец спринта | Демонстрация функциональности аналитику |

| Приёмка | Аналитик подтверждает: "Готово" / "Требует доработки" |

| После приёмки | Фиксация версии в репозитории, обновление документации |

| Hotfix | При критической ошибке — срочный релиз в течение дня |

## 6.7 Выходные документы по стадиям

| Стадия | Документ |

||-|

| Формирование ТЗ | Техническое задание (настоящий документ) |

| Реализация | Исходный код, `requirements.txt`, `models/` |

| Тестирование | Программа и методика испытаний (ПМИ), отчёт о приёме |

| Внедрение | Руководство пользователя, руководство администратора |

| Сопровождение | Журнал изменений, список известных проблем |

## 6.7 Изменения

| Что | v0.1 | v0.2 |

|-|||

| Количество разделов | 10 | 6 |

| Объём | ~50 страниц | ~25–30 страниц |

| Структура | Классическая ГОСТ-структура | Оптимизирована под Agile и MVP |

| Удалено | - Условия хранения и эксплуатации<br>- Гарантийные обязательства<br>- Подробный перечень документации<br>- Требования к актуализации<br>- Инструменты разработки | Все удалено как избыточное для внутреннего ПО |

| Объединено | Разделы: Надёжность, Условия эксплуатации, Защита информации, Техническое обеспечение, Стадии разработки, Документация | Сведено в 3 логичных раздела: Нефункциональные требования, Архитектура, Жизненный цикл |

| Добавлено | — | - Раздел "Ограничения и допущения" (в будущем)<br>- Глоссарий (в будущем)<br>- Roadmap (в будущем)<br>- Явное разделение на Task и User Story в плане спринтов |

| Фокус | На формальном согласовании | На практической реализации |

✅ Что хорошо добавлено:

- Чёткая привязка задач к `product\_backlog.xlsx`

- Явное разделение на User Story и Task

- Учёт предсказания атрибутов моделью

- Учёт таблицы `contractors` и смены сети

- Удаление избыточных ГОСТ-формальностей