Описание функциональных характеристик программного обеспечения

Платформа «Панда»

Наименование ПО	Платформа «Панда»
Правообладатель	ООО «НЕЙРОПРЕМ»
ИНН / ОГРН / КПП	1684031969 / 1261600019537 / 168401001
Адрес	420141, Республика Татарстан, г. Казань, ул. Кул Гали, д. 7Б, к. 1, кв. 53
Руководитель	Хисамов Тимур Ринатович (директор, единственный учредитель — 100%)
ОКВЭД	62.01
Контакты	info@neuroprem.ru, +7 995 007-17-57
Сайт ПО	https://panda-ai.ru
Дата документа	01.08.2026

1. Назначение программного обеспечения

Платформа «Панда» — программное обеспечение для **автоматизации обработки входящих обращений**
и ведения диалогов с клиентами в мессенджерах с использованием технологий обработки естественного
языка (NLP) и больших языковых моделей (LLM).

ПО решает следующие задачи заказчика:

1. **Приём входящих обращений** из мессенджеров и с веб-сайта в единую точку обработки.
2. **Автоматическое ведение диалога** на естественном русском языке по заданной оператором инструкции (системному промпту) и загруженным справочным материалам.
3. **Квалификация обращения** — структурированное извлечение из переписки признаков потребности, срочности и бюджета (фреймворк NUB: Need / Urgency / Budget).
4. **Маршрутизация обращения** — перевод диалога в режим ручной обработки оператором, постановка на паузу, закрытие.
5. **Передача структурированных данных обращения во внешние учётные системы** заказчика
   (CRM-системы, складские системы) и по исходящему веб-хуку.
6. **Учёт и анализ** обработанных обращений, потребления вычислительных ресурсов.

Область применения: организации малого и среднего бизнеса, обрабатывающие входящий поток обращений в мессенджерах (услуги, розничная торговля, медицина, образование, недвижимость, автосервис и иные отрасли — в ПО предусмотрен 21 отраслевой преднастроенный шаблон

инструкции).

Вид ПО: прикладное программное обеспечение, поставляемое как по модели удалённого доступа (SaaS), так и в варианте установки в инфраструктуре заказчика (on-premise) — архитектура ПО (контейнеризация, отсутствие зависимости от внешних управляющих сервисов) допускает оба варианта.

> **Примечание к формулировкам.** В настоящем документе и в карточке реестра назначение ПО описывается нейтрально: обработка обращений, ведение диалога, квалификация, маршрутизация,
> передача данных во внешние системы. ПО **не осуществляет** поиск информации о потенциальных
> покупателях (продавцах) во внешних источниках, не размещает предложения на площадках и не обеспечивает заключение сделок — эта функциональность в составе ПО отсутствует.

---

2. Класс программного обеспечения

ПО относится к **прикладному программному обеспечению**. Рекомендуемый класс по приказу 486:
- **основной — 05.09** «Средства управления диалоговыми роботами (чат-боты, голосовые роботы)» (под этим классом в реестре зарегистрированы аналогичные продукты, например AutoFAQ — реестр № 7394);
- **дополнительные — 09.09** «Средства управления отношениями с клиентами (CRM)» и **05.08** «Средства управления контактными центрами».

Дополнительно целесообразно запросить признак **«ПО в сфере искусственного интеллекта»**. Точный
код класса подтверждается по действующему классификатору при подаче заявления.

Обоснование класса:
- Класс должен соответствовать фактическому назначению — средства автоматизации обработки обращений / взаимодействия с клиентами (близко к CRM-системам и системам управления процессами организации).
- **Не следует** выбирать класс, относящийся к «PaaS на генеративных моделях»: согласно ПП 1937 для таких классов предъявляются требования к ЦОД (от 10 МВт) и парку GPU (от 1000), которым ООО «НЕЙРОПРЕМ» не соответствует. Платформа «Панда» — прикладное ПО, использующее
внешние LLM-сервисы как поставщиков, а не платформа для предоставления генеративных моделей.

---

3. Функциональные характеристики

Ниже описана функциональность, **фактически реализованная в коде** текущей версии. Ограничения и нереализованные возможности вынесены в раздел 9.

3.1. Управление организацией, пользователями и правами

- Мультиарендная (multi-tenant) модель: каждая организация-заказчик — отдельный «тенант»; данные тенантов изолированы на уровне приложения (фильтрация по `tenant_id` в каждом обработчике + проверка принадлежности ресурса).
- Аутентификация и авторизация — через OIDC-провайдер Keycloak (realm `panda`), протокол OAuth 2.0 / OpenID Connect, поток Authorization Code + PKCE (S256), подпись токенов RS256, проверка на стороне сервисов через JWKS.
- Ролевая модель из четырёх ролей с иерархией прав: `viewer` (0) → `operator` (1) → `admin` (2) → `owner` (3). Отдельная платформенная роль `platform_admin` для администратора платформы (реализована как realm-роль Keycloak).
- Приглашение сотрудников в организацию по e-mail, назначение и изменение роли, удаление участника. Принятие приглашения требует подтверждённого адреса электронной почты (защита от перехвата доступа к чужой организации).
- Онбординг: пошаговое первичное заполнение данных организации.
- Возможность блокировки (заморозки) тенанта администратором платформы.

Реализующие компоненты: `apps/api-gateway/src/auth.py`, `provisioning.py`, `routers/settings.py`, `routers/admin.py`, `infra/keycloak/realm-export.json`.

3.2. Конфигурирование диалоговых агентов

«Агент» — именованная конфигурация автоматической обработки обращений. Настраиваются:

Группа	Параметры
Основное	наименование, описание, аватар, признак активности, часовой пояс, язык
Инструкция	системный промпт (лимит 8000 символов), первое приветственное сообщение (1000), отраслевой преднастроенный шаблон (21 шт.)
Модель	температура (стиль ответа), максимальное число токенов, penalties, признаки обработки аудио/изображений/файлов
Поведение	задержка ответа (`wait_time`), разбиение ответа на абзацы и пауза между ними, глубина истории диалога (в сообщениях и по времени)
Управление	фразы-триггеры паузы и возобновления диалога, статус нового диалога по умолчанию
Ограничения	лимит сообщений в пробном режиме (по умолчанию 100)
Расписание	режим работы по дням недели и интервалам времени
Защита от спама	порог количества сообщений за интервал, сообщение-ответ

- Генерация проекта инструкции по текстовому описанию бизнеса (`POST /agents/prompt-suggest`).
- Пользовательские функции агента (`agent_functions`) — вызываемые моделью инструменты.

Реализующие компоненты: `apps/api-gateway/src/routers/agents.py`, `agent_settings.py`; миграции `0002`, `0003`, `0017`.

3.3. База знаний (поиск по документам заказчика, RAG)

Функциональный блок, обеспечивающий ответы агента на основе загруженных заказчиком материалов
(прайс-листы, описания услуг, регламенты).

- **Загрузка документов** через веб-интерфейс. Поддерживаемые форматы: **PDF, DOCX, TXT, MD, CSV**.
- **Ограничение размера файла — 20 МБ** (превышение отклоняется с кодом HTTP 413).
- **Хранение оригиналов** — в объектном S3-совместимом хранилище (MinIO), бакет `kb-documents`,
  ключ вида `{agent_id}/{document_id}{расширение}`.
- **Извлечение текста:** PDF — библиотека `pypdf`; DOCX — `python-docx` (абзацы);
  TXT/MD/CSV — чтение как текст в кодировке UTF-8.
- **Разбиение на фрагменты:** фиксированное окно 1000 символов с перекрытием 150 символов.
- **Векторизация:** сервис Yandex Foundation Models, эндпоинт
  `https://llm.api.cloud.yandex.net/foundationModels/v1/textEmbedding`.
  Используются **две парные модели**: `text-search-doc` — для фрагментов документов при индексации, `text-search-query` — для поискового запроса. **Размерность вектора — 256.**
- **Хранение векторов:** PostgreSQL с расширением pgvector, тип `vector(256)`,
  индекс **HNSW** по косинусной метрике (`vector_cosine_ops`, `m=16`, `ef_construction=64`).
- **Поиск:** при обработке обращения выполняется векторный поиск ближайших фрагментов (косинусное расстояние), найденные фрагменты передаются модели в составе контекста.
- **Асинхронная индексация:** документ ставится в очередь Redis Streams (`panda:kb_ingest`, consumer group `kb-ingest`), обрабатывается фоновым потребителем в составе agent-engine. Статусы документа отображаются в интерфейсе: «Обработка» / «Готов» / «Ошибка».
- Просмотр списка документов и удаление документа (с удалением фрагментов и вектора).

Реализующие компоненты: `apps/api-gateway/src/routers/knowledge.py`,
`apps/agent-engine/src/ingest.py`, `apps/agent-engine/src/rag.py`,
`packages/shared-python/kycai_shared/models/knowledge.py`; миграция `0019`.

3.4. Каналы приёма обращений

Основные каналы — **российские и нейтральные мессенджеры**. Продукт полнофункционален без каких-либо иностранных сервисов:

Канал	Способ подключения	Реализация
Telegram	токен бота (@BotFather); приём обновлений методом `getUpdates` по защищённому исходящему каналу	`connector-service/src/main.py`, `telegram_polling.py`, `outbound.py`
ВКонтакте	ключ доступа сообщества + идентификатор сообщества; приём — VK Bots Long Poll (сообщество конфигурируется автоматически)	`connector-service/src/vk_polling.py`
Авито	авторизация OAuth (client credentials); приём — по веб-хуку на публичный домен	`connector-service/src/main.py`
MAX (российский мессенджер)	токен бота (@MasterBot); приём — по веб-хуку	то же
Чат на сайте (веб-виджет)	встраиваемый JS-виджет, отдаётся платформой; обмен по HTTP	`connector-service/src/widget.py`

Дополнительно, **по выбору заказчика**, поддерживается подключение WhatsApp через стороннего провайдера (GreenAPI) — опциональная интеграция, не входящая в основную функциональность; работоспособность продукта от неё не зависит.

- Приём событий — по веб-хукам/опросу на выделенный публичный домен (`hooks.*`).
- Подлинность входящих проверяется криптографически (Telegram — сверка секрета через `hmac.compare_digest`).
- Учётные данные каналов **хранятся в зашифрованном виде** (см. 3.11).

3.5. Обработка обращения и ведение диалога

- Ядро обработки — конечный автомат (граф состояний) на LangGraph.
  Состояние диалога сохраняется в PostgreSQL (checkpointer `AsyncPostgresSaver`, отдельная схема `langgraph`), идентификатор потока `thread_id` = идентификатор диалога. Это обеспечивает сохранение контекста диалога при перезапуске сервиса и изоляцию диалогов.
- Узлы графа: `analyze` (единый вызов модели — классификация намерения и извлечение признаков
  NUB) → маршрутизация на `qualify_lead` / `respond_general` / `transfer_human`.
- **Квалификация NUB** — извлечение из переписки признаков: Need (потребность), Urgency (срочность), Budget (бюджет); результат сохраняется в состоянии диалога и отображается оператору.
- Обмен между сервисами — асинхронный, через Redis Streams:
  `panda:messages` (входящие), `panda:outbound` (исходящие), `panda:notifications`, `panda:billing_events`, `panda:kb_ingest`. Реализованы consumer groups, подтверждение обработки, очередь недоставленных сообщений (DLQ) после 3 повторов.
- **Транзакционный outbox** (`outbox`) — гарантия доставки исходящих сообщений и сохранение их порядка при разбиении ответа на абзацы.
- Защита от инъекций в промпт — единый guard на всех путях обращения к модели.

- Учёт лимита сообщений пробного режима для тенантов без действующей подписки.
- **Инструменты агента (вызов функций моделью).** Помимо пользовательских функций (раздел 3.2) агенту автоматически становятся доступны встроенные инструменты — при условии,
  что у организации включена соответствующая интеграция и заданы корректные учётные данные:

Инструмент	Назначение	Условие появления
`get_time` / `get_date`	текущие дата и время	всегда
`transfer_to_human`	передача диалога оператору	всегда
`crm_create_lead`	создание сделки в CRM	активная интеграция CRM
`send_email`	письмо оператору (не более 3 в час на диалог)	всегда
`send_sms`	SMS клиенту (не более 3 в час на диалог)	интеграция SMS
`create_payment_link`	ссылка на оплату	интеграция ЮKassa
`create_calendar_event`	запись встречи в календарь (не более 3 в час на диалог)	интеграция «Яндекс Календарь»
`catalog_search`	поиск товара или услуги с актуальной ценой и остатком	интеграция «1С: обмен с сайтом»
`yclients_*` / `altegio_*`	услуги, свободное время, запись клиента	интеграция YClients или Altegio

Сведения о ценах и наличии предоставляются агенту исключительно указанным инструментом (обращение к данным, полученным из учётной системы заказчика), а не средствами поиска по документам, — это исключает выдачу устаревших сведений.

Реализующие компоненты: `apps/agent-engine/src/graphs/` , `worker.py` , `functions.py` ; ADR-002, ADR-004.

3.6. Передача данных во внешние системы

Система	Реализованная операция	Способ подключения
amoCRM	создание сделки (lead) и примечания, REST API v4	долгоживущий токен доступа
Bitrix24	создание лида (`crm.lead.add`)	URL входящего веб-хука
RetailCRM	создание заказа, API v5	ключ API
МойСклад	создание контрагента	ключ API
Исходящий веб-хук	событие `lead.qualified` с подписью **HMAC** в заголовке `X-Panda-Signature`	URL + секрет
ЮKassa (в диалоге)	формирование ссылки на оплату по ключам **магазина заказчика**	ключи магазина заказчика
SMS (SMS Aero / SMS.RU)	отправка SMS как инструмент агента (ограничение 3/час)	ключ API
YClients / Altegio	работа с записями и профилями	ключ API
Яндекс Календарь	создание события в календаре организации как инструмент агента `create_calendar_event` (ограничение 3/час на диалог), протокол **CalDAV** (RFC 4791), формат iCalendar (RFC 5545)	логин Яндекса + пароль приложения
1С (УТ, УНФ, Розница)	приём выгрузки номенклатуры, цен и остатков штатным механизмом **«Обмен с сайтом»** (формат **CommerceML** 2.x/3.x); данные доступны агенту инструментом `catalog_search` ; встречная выдача заказов по запросу `type=sale`	адрес обмена, логин и пароль, формируемые платформой; настройка выполняется штатным мастером 1С без доработки конфигурации

- Все исходящие HTTP-вызовы к внешним системам защищены от SSRF (проверка адресов назначения).
- Подключение CRM выполняется **вводом ключа/токена/URL** (OAuth-поток авторизации в интерфейсе платформы не реализован — см. раздел 9).
- Обмен с 1С организован по инициативе информационной базы заказчика (исходящие соединения
  от 1С к платформе), что не требует публикации базы 1С в сети Интернет и наличия у заказчика
  постоянного внешнего IP-адреса. Приём выгрузки защищён отдельными для каждой интеграции
  учётными данными, ограничением объёма передаваемых файлов и контролем имён файлов архива.

Реализующие компоненты: `apps/agent-engine/src/crm.py` (реестр адаптеров `CRM_ADAPTERS`), `apps/agent-engine/src/caldav_yandex.py` (Яндекс Календарь), `apps/connector-service/src/one_c.py` и `apps/connector-service/src/commerceml.py` (обмен с 1С), `apps/api-gateway/src/routers/integrations.py` .

3.7. Рабочее место оператора

- Список диалогов агента с постраничной навигацией и фильтрацией; карточка диалога с историей сообщений.
- **Ручной ответ оператора клиенту** из интерфейса платформы (сообщение уходит в исходную переписку через соответствующий канал).
- Изменение статуса диалога (в т.ч. постановка на паузу — при статусе `paused` автоматические ответы агента не отправляются).
- Обновление интерфейса в реальном времени — через WebSocket (Centrifugo; сервер выдаёт клиенту HMAC-подписанные токены подключения и подписки).
- **Удаление диалога** с удалением связанных сообщений и состояний LangGraph — реализация требования об уничтожении персональных данных (152-ФЗ).

Реализующие компоненты: `apps/api-gateway/src/routers/conversations.py`, `apps/web/src/pages/agents/[agentId]/threads/`, `routers/ws.py`.

3.8. Тестовый чат

Проверка настроенного агента без подключения канала: изолированный поток вида `test-{agent_id}-{session}`, отображение результата квалификации (панель «Анализ агента»). Реализация: `apps/api-gateway/src/routers/agent_test_chat.py` (`POST /agents/{id}/test-chat`).

3.9. Аналитика и отчётность

- Сводная панель (`GET /dashboard`): агрегированные показатели по организации.
- Аналитика (`GET /analytics?days=N`): динамика за период.
- Потребление токенов (`GET /billing/usage`): ряд по дням за 14 суток.
- Графические панели Grafana для эксплуатации (см. документ 03).

3.10. Биллинг и учёт потребления

- Тарифные планы хранятся в БД (таблица `billing_plans`), отдаются интерфейсу (`GET /billing/plans`) с фильтром по признаку активности.
- Оплата — через платёжный сервис **ЮKassa** (резидент РФ). Формирование платежа (`POST /billing/checkout`), страница оплаты, счета (`invoices`).
- **Фискализация по 54-ФЗ**: в платёж передаётся состав чека (`receipt`), ставка НДС настраивается параметром `YOOKASSA_VAT_CODE`.
- **Обработка уведомлений об оплате** — веб-хук ЮKassa. Реализованы два независимых контроля: проверка IP-адреса источника по документированным диапазонам ЮKassa **и** повторный запрос статуса платежа в API ЮKassa со сверкой суммы и валюты (ЮKassa не подписывает тело уведомления). Активация подписки идемпотентна (блокировка строки `SELECT ... FOR UPDATE`).
- **Учёт потребления токенов**: запись в `usage_records` и атомарное увеличение счётчика подписки на стороне БД; уведомления при достижении 80% и 100% лимита.
- Расчётный период — 30 дней с момента подтверждённой оплаты.
- Возврат платежа реализован в billing-service (идемпотентно, с проверкой принадлежности тенанту) — см. ограничения в разделе 9.

Реализующие компоненты: `apps/billing-service/src/payments.py`, `main.py`, `apps/agent-engine/src/usage.py`, `apps/api-gateway/src/routers/billing.py`.

3.11. Защита информации и журналирование

Механизм	Реализация
Шифрование учётных данных	Учётные данные каналов, CRM и пользовательские ключи LLM шифруются **Fernet** (AES-128-CBC + HMAC) через `kycai_shared.crypto.EncryptedJSON`. Поддержана ротация ключей (`MultiFernet`, `ENCRYPTION_KEYS`)
Журнал действий	Таблица `audit_log`: действие, тип и идентификатор ресурса, прежнее и новое значение, IP, User-Agent, время. Пишется отдельной транзакцией — сбой журналирования не откатывает основную операцию
Разграничение доступа	RBAC (`require_role`), проверка принадлежности ресурса тенанту в каждом обработчике
Межсервисное взаимодействие	Заголовок `X-Internal-Token`, fail-closed при пустом значении
Ограничение частоты запросов	slowapi, 120 запросов/мин по умолчанию; для публичных методов — 10–15/мин; хранилище счётчиков — Redis
Заголовки безопасности	`X-Content-Type-Options`, `X-Frame-Options: DENY`, `Referrer-Policy`, `Permissions-Policy`, HSTS (в промышленной среде)
Контроль конфигурации	Отказ запуска в режиме `production` при незаданных `ENCRYPTION_KEY` / `INTERNAL_API_TOKEN`, при `TRUSTED_PROXY_HOSTS='*'`, а также при совпадении секретов со списком известных значений по умолчанию
Транспорт	TLS терминируется на обратном прокси Caddy, сертификаты Let's Encrypt; административный API прокси отключён

Журналируемые действия: создание и удаление агента, создание и удаление канала, изменение роли и удаление участника, завершение онбординга, изменение тенанта администратором платформы,
операции партнёрской программы.

---

4. Архитектура и состав ПО

Платформа реализована как набор независимо развёртываемых сервисов (монорепозиторий, ADR-001), взаимодействующих через HTTP и Redis Streams.

Компонент	Назначение	Порт
`api-gateway`	публичный REST API, аутентификация, авторизация, бизнес-логика кабинета	8000
`agent-engine`	исполнение графа диалога (LangGraph), RAG, интеграции, планировщик	8001
`connector-service`	приём веб-хуков каналов, отправка исходящих сообщений, веб-виджет	8002
`billing-service`	платежи и подписки (ЮKassa)	8003
`notification-service`	уведомления по электронной почте	8004
`web`	одностраничное веб-приложение личного кабинета (React)	—
`site`	публичный сайт (Next.js, статическая генерация)	3000
`packages/shared-python` (`kycai_shared`)	общий код: конфигурация, доступ к БД, миграции, шифрование, журнал, шина событий, метрики	—

> Внутренний Python-пакет общего кода носит историческое техническое имя `kycai_shared`. Это имя библиотеки внутри монорепозитория; оно **не связано с каким-либо внешним сервисом, сторонним поставщиком или зарубежной зависимостью**. Весь код пакета разработан ООО «НЕЙРОПРЕМ» и входит в состав платформы «Панда».

Хранилища и инфраструктурные компоненты: PostgreSQL 16 + pgvector (основная БД и векторный
поиск), Redis (кеш, шина событий, счётчики), MinIO (объектное хранилище документов), Keycloak (OIDC), Centrifugo (WebSocket), ClickHouse (аналитика), Caddy (обратный прокси, TLS), Prometheus / Grafana / Alertmanager (мониторинг).

Модель данных: 32 таблицы, схема управляется Alembic (22 миграции, линейная цепочка `0001` → `0022`). Таблица `messages` секционирована по месяцам (RANGE-партиционирование). Значение поставщика LLM по умолчанию для новых агентов — `yandexgpt` (миграция `0022`).

Объём собственного кода: приблизительно 18 200 строк на Python и 13 900 строк на TypeScript/TSX (без учёта зависимостей, сгенерированных файлов и исследовательских материалов).

---

5. Используемые языковые модели и резидентность данных

Роль	Поставщик	Модель
Основная модель генерации	Yandex Foundation Models (ООО «Яндекс.Облако», РФ)	`yandexgpt` / `yandexgpt-lite`
Резервная модель генерации	GigaChat (ПАО «Сбербанк», РФ)	через клиент `langchain-gigachat`
Векторизация базы знаний	Yandex Foundation Models (РФ)	`text-search-doc`, `text-search-query`, размерность 256

- Вызов YandexGPT — через **OpenAI-совместимый эндпоинт Yandex Cloud** (`https://llm.api.cloud.yandex.net/v1`), поэтому в качестве транспорта используется открытая библиотека OpenAI-протокола (сам сервис OpenAI не вызывается — см. техстек-документ). Аутентификация: сервисный API-ключ (заголовок `Api-Key`) либо IAM-токен; GigaChat — OAuth, обновление токена выполняет клиентская библиотека `langchain-gigachat`.
- Порядок автоматического переключения при недоступности основного поставщика: `yandexgpt` → `gigachat`. Провайдер участвует в цепочке только при заданном ключе.
- **Обработка данных осуществляется на инфраструктуре и сервисах, размещённых в Российской Федерации.** Поддерживаются **исключительно поставщики — резиденты РФ** (`yandexgpt`, `gigachat`). Иностранные поставщики (OpenAI, Anthropic, DeepSeek, OpenRouter) **из состава ПО удалены** и в клиент не разрешаются; персональные данные в промптах зарубежным поставщикам не передаются (152-ФЗ ст.12).

Роутер языковых моделей (`apps/agent-engine/src/llm/router.py`) допускает только `yandexgpt` и `gigachat`; запрос любого иного поставщика возвращает ошибку «unsupported LLM provider». Соответствие проверяется регрессионными тестами `test_llm_router.py` (в частности `test_foreign_providers_not_supported`, `test_auto_fallback_is_rf_only`).
- Хранилища персональных данных (PostgreSQL, MinIO, Redis, ClickHouse) размещаются в дата-центрах на территории РФ (ст. 18.5 152-ФЗ).
- ООО «НЕЙРОПРЕМ» внесено в реестр операторов персональных данных Роскомнадзора, регистрационный номер оператора **16-26-064484**.

Отсутствие принудительного управления из-за рубежа: ПО не содержит механизмов принудительного обновления, удалённой деактивации или управления со стороны иностранных лиц. Обновление выполняется администратором платформы (или заказчиком в варианте on-premise) самостоятельно. Техническая поддержка оказывается на территории РФ.

---

6. Состав программного обеспечения и права

6.1. Права на платформу

**Исключительные права на платформу «Панда» — код всех сервисов (`apps/` , `packages/`), модель данных, миграции, конфигурации развёртывания, пользовательский интерфейс, отраслевые
шаблоны инструкций и документацию — принадлежат ООО «НЕЙРОПРЕМ» в полном объёме.**

Права на результаты разработки закреплены за ООО «НЕЙРОПРЕМ» на основании трудовых отношений
с работниками (служебные произведения, ст. 1295 ГК РФ) и договоров с привлечёнными исполнителями
(при их наличии) с условием об отчуждении исключительных прав.

6.2. Используемые компоненты с открытым исходным кодом

Платформа использует свободное ПО, распространяемое на условиях разрешительных лицензий. Все компоненты используются **без модификации**, как отдельные библиотеки или самостоятельные сервисы; производные произведения на их основе не создавались.

Основные библиотеки (Python):

Компонент	Назначение	Лицензия
FastAPI	веб-фреймворк REST API	MIT
Uvicorn	ASGI-сервер	BSD-3-Clause
Pydantic / pydantic-settings	валидация данных и конфигурации	MIT
SQLAlchemy	ORM и доступ к БД	MIT
Alembic	миграции схемы БД	MIT
asyncpg	драйвер PostgreSQL	Apache-2.0
pgvector (клиент)	работа с векторным типом	MIT
redis-py	клиент Redis	MIT
httpx	HTTP-клиент	BSD-3-Clause
structlog	структурированное журналирование	Apache-2.0 / MIT
cryptography	шифрование (Fernet)	Apache-2.0 / BSD-3-Clause
LangGraph	конечный автомат диалога	MIT
LangChain (core, community)	абстракции работы с LLM	MIT
langchain-gigachat	клиент GigaChat	MIT
pypdf	извлечение текста из PDF	BSD-3-Clause
python-docx	извлечение текста из DOCX	MIT
slowapi	ограничение частоты запросов	MIT
minio	клиент S3-совместимого хранилища	Apache-2.0
yookassa	SDK платёжного сервиса	MIT
OpenTelemetry SDK	телеметрия	Apache-2.0

Основные библиотеки (Frontend): React 19 (MIT), TypeScript (Apache-2.0), Vite (MIT), Tailwind CSS v4 (MIT), Radix UI (MIT), TanStack Query (MIT), React Router (MIT), React Hook Form (MIT), Zod (MIT), Zustand (MIT), Framer Motion (MIT), keycloak-js (Apache-2.0), centrifuge-js (MIT), Next.js (MIT).

Инфраструктурные компоненты (используются как самостоятельные сервисы в контейнерах):

Компонент	Назначение	Лицензия
PostgreSQL	СУБД	PostgreSQL License
pgvector (расширение)	векторный поиск	PostgreSQL License
Redis	кеш, шина событий	см. примечание
MinIO	объектное хранилище	**AGPL-3.0**
Keycloak	OIDC-провайдер	Apache-2.0
Centrifugo	сервер WebSocket	MIT
ClickHouse	аналитическая СУБД	Apache-2.0
Caddy	обратный прокси, TLS	Apache-2.0
Prometheus	сбор метрик	Apache-2.0
Grafana	панели мониторинга	AGPL-3.0
Alertmanager	оповещения	Apache-2.0

Примечания к составу:

1. **MinIO (AGPL-3.0) и Grafana (AGPL-3.0)** используются как немодифицированные самостоятельные сетевые сервисы; их исходный код не изменялся и в состав платформы «Панда» не включается — они являются компонентами среды развёртывания. Условия AGPL не распространяются на код платформы, поскольку производное произведение не создаётся. При необходимости MinIO заменяется любым S3-совместимым хранилищем (взаимодействие ведётся по стандартному протоколу S3).
2. **Redis** — лицензия зависит от версии: до 7.4 — BSD-3-Clause, начиная с 7.4 — RSALv2 / SSPL. Конкретная используемая версия зафиксирована в файлах развёртывания; компонент используется без модификации как самостоятельный сервис.
3. **GigaChat и YandexGPT** используются **как облачные сервисы по API** российских поставщиков (ПАО «Сбербанк» и ООО «Яндекс.Облако» соответственно), а не как самостоятельно размещаемые модели. В состав ПО веса моделей не входят. Открытые веса GigaChat
   (лицензия MIT) допускают вариант локального размещения — в текущей версии он не реализован.
4. **Критическая зависимость от иностранных компонентов отсутствует:** все используемые открытые библиотеки распространяются на разрешительных лицензиях без права отзыва, поставляются из локально зафиксированных версий (`uv.lock` , `pnpm-lock.yaml`) и при необходимости могут быть заменены. Поставщики LLM и платёжный сервис — резиденты РФ.
   Выплаты иностранным лицам от реализации ПО отсутствуют.

Полная опись зависимостей с указанием лицензий формируется из файлов фиксации версий (`uv.lock` , `pnpm-lock.yaml`) и прилагается к заявке; таблица выше перечисляет основные компоненты.

---

7. Системные требования

Подробно — в документе `03-Руководство-администратора.md`.
Кратко: серверная ОС семейства Linux с поддержкой Docker и Docker Compose; развёртывание всех компонентов — в контейнерах; работа с ПО пользователем — через современный веб-браузер. Действующая промышленная среда работает на виртуальном сервере с 6 ГБ ОЗУ (Ubuntu 24.04).

Требование ПП 1937: совместимость не менее чем с двумя российскими операционными системами
(Astra Linux, РЕД ОС, ALT Linux). Платформа — веб-приложение с серверной частью в контейнерах Linux; для подтверждения достаточно протокола тестирования в браузерах указанных ОС. Протоколы
совместимости оформляются и предоставляются дополнительно.

---

8. Сведения о стоимости

ПО распространяется **свободно на территории Российской Федерации** — доступ приобретается любым лицом через сайт https://panda-ai.ru на условиях публичной оферты (https://panda-ai.ru/offer). Ограничений по кругу лиц нет.

Позиция	Стоимость
Пробный период (7 дней: 1 агент, 1 канал, 100 сообщений, без банковской карты)	0 ₽
Подключение (первый месяц использования включён)	10 000 ₽
Сопровождение (со 2-го месяца)	5 999 ₽ / мес
Потребление токенов языковых моделей	по фактическому расходу (ориентировочно 1–3 ₽ за диалог)

Оплата — в рублях РФ через ЮKassa.

Реализация прав использования ПО осуществляется по **лицензионному договору** (простая неисключительная лицензия на удалённый доступ): подключение — паушальная часть, сопровождение —
периодическая, потребление токенов — переменная часть лицензионного вознаграждения. После включения ПО в реестр реализация прав освобождается от НДС (пп. 26 п. 2 ст. 149 НК РФ).

---

9. Ограничения текущей версии

Раздел приведён для полноты и достоверности описания. Перечисленная ниже функциональность **не заявляется как реализованная**.

Функция	Фактическое состояние
Коннекторы **Zapier / Make**	В интерфейсе присутствуют, но не функционируют: форма сохраняет параметр `webhook_url`, адаптер читает `url` — несовпадение имён приводит к отсутствию вызова. Не заявляются. Задача решается универсальным исходящим веб-хуком (см. 3.6)
Проставление тегов диалогам	Реализован только справочник тегов и их отображение. Автоматическое присвоение тегов диалогу отсутствует (движок не записывает `conversation.tags`). Не заявляется
Автоматические напоминания (follow-up)	Планировщик (`apps/agent-engine/src/followup.py`), интерфейс и переключатель активации реализованы. Функция выключена по умолчанию и **на дату составления документа не подтверждена сквозной проверкой** — до подтверждения не заявляется
Автоматическая передача оператору	**Реализована частично и имеет дефект.** Узел `transfer_human` возвращает клиенту сообщение о передаче, выставляет признак `should_transfer`, переводит квалификацию в стадию `transferred` и однократно публикует уведомление оператору (доходит до отправки письма). Однако статус диалога при этом **не переводится в** `paused`, поэтому агент продолжает автоматически отвечать на последующие сообщения клиента параллельно с оператором. До устранения дефекта функция **не заявляется**; заявляется только **ручное вмешательство оператора** (п. 3.7), которое подтверждено
Каналы Instagram, Wazzup	Обозначены в интерфейсе как «Скоро», отключены. Не реализованы
Интеграции Google Таблицы, Notion, календари	Обозначены как «Скоро», отключены. Не реализованы
OAuth-подключение CRM	Не реализовано ни для одной CRM. Подключение — вводом токена/ключа/URL
Обновление сделок в CRM	Реализовано только создание. Обновление существующих сделок отсутствует
Загрузка базы знаний **по ссылке (URL)**	Не реализована
Повторная индексация документа базы знаний	Отдельная операция отсутствует; при ошибке требуется повторная загрузка файла
Извлечение **таблиц из DOCX**	Не реализовано (обрабатываются только абзацы). Для прайс-листов рекомендуются форматы CSV/TXT
Рекуррентные (автоматические) списания	Не реализованы. Поле `yookassa_subscription_id` в схеме не используется. Продление — по факту нового платежа
Возврат платежа из интерфейса	Реализован в billing-service, но метод не опубликован в API-шлюзе; выполняется администратором платформы

Функция	Фактическое состояние
Распознавание речи (voice)	Не реализовано; зависимость `openai-whisper` вынесена в необязательную группу и не устанавливается
Разграничение на уровне строк БД (RLS)	Не применяется; изоляция тенантов обеспечивается на уровне приложения

---

10. Ссылки на сопутствующие документы

Документ	Содержание
`02-Руководство-пользователя.md`	работа в личном кабинете
`03-Руководство-администратора.md`	установка, эксплуатация, резервное копирование
`04-Процессы-разработки-и-поддержки.md`	жизненный цикл, поддержка, SLA
`05-Инструкция-по-установке-экземпляра-для-экспертизы.md`	развёртывание проверочного экземпляра

Архитектурные решения зафиксированы в ADR: `panda-dev/docs/architecture/adr/001…005`.