Практическая архитектура AI поддержки: как развернуть безопасный чатбот в малом бизнесе

Олег Зельдин
Апекс Берг

Безопасный чатбот — это не «умная переписка»

Главный вопрос для бизнеса: не может ли бот ответить,
а можем ли мы управлять тем, как он отвечает.

Тема доклада — не выбор сильнейшей языковой модели,
а проектирование безопасного и управляемого контура ИИ-поддержки.

Малый бизнес

Ограниченные бюджеты

Дефицит ИИ-инженеров

Потребность в быстром запуске

Общие архитектурные вызовы

Контакт-центр

Масштаб операций

Сложные регламенты

Высокие репутационные риски

Необходимость строгих границ работы

Контроль над внутренней базой знаний

Исключение риска работы от имени компании без надзора


Откуда модель берет знания?
Какие действия запрещены?
«Клиент»
«Модель»
«Ответ»
Когда происходит эскалация на человека?
Слишком простой контур. Не жизнеспособен в операционной среде.

«Внедрять надо не самого умного бота, а управляемую архитектуру ИИ-поддержки»

Модель сама по себе уязвима. Безопасноа. Безопасность обеспечивают слои контроля: база знаний, права доступа, журналирование и маршрутизация.

«Потребность бизнеса»

- Автоматизация рутинной поддержки клиентов.
- Доступ к внутренней базе знаний и регламентам.
- Мгновенные и точные ответы.

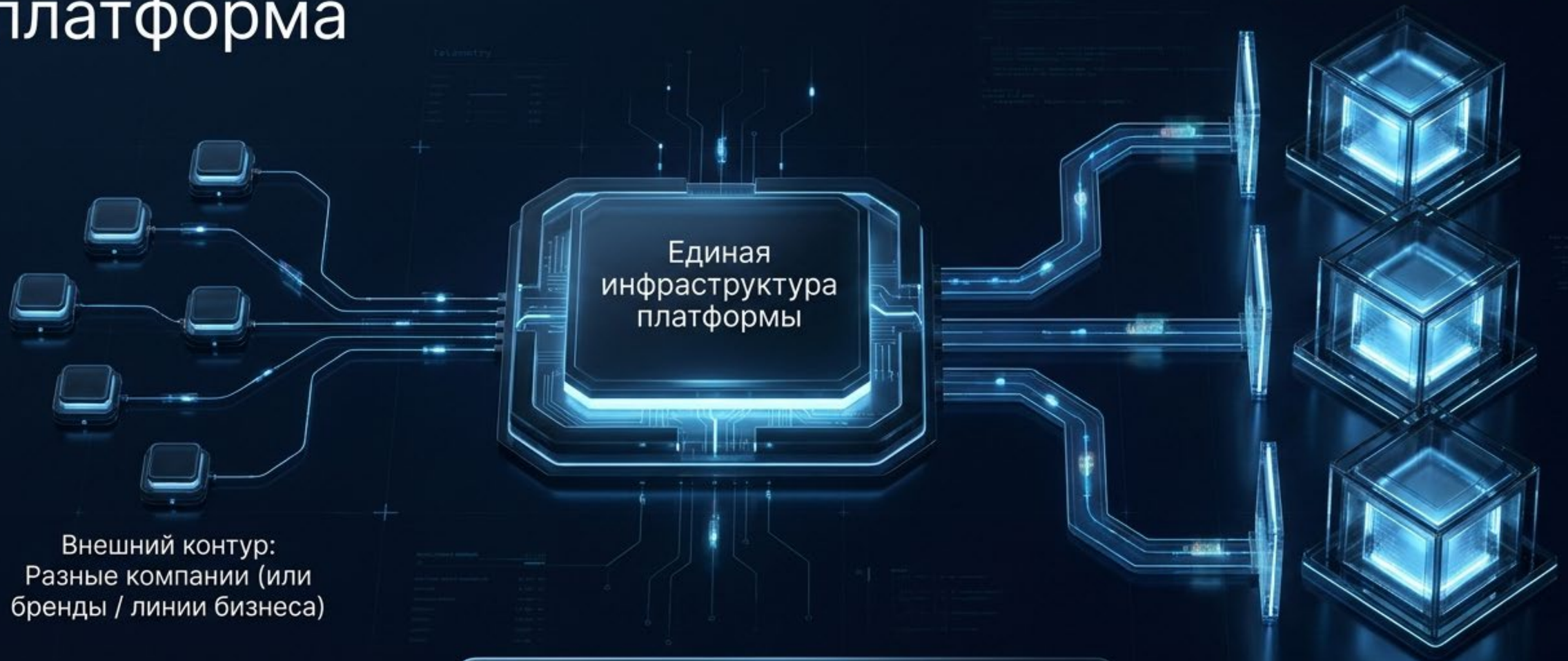
Инфраструктурные ограничения

- Отсутствие бюджета на тяжелые enterprise-решения.
- Нехватка разработчиков и отдела ИИ-безопасности.
- Сложности с техническим сопровождением.

Задача: дать рабочий инструмент, не потеряв контроль над данными и поведением.

Решение: многоарендная платформа

Изолированные хранилища:
Индивидуальные базы знаний
и контексты чатботов



Внешний контур:
Разные компании (или
бренды / линии бизнеса)

Бот должен видеть только те данные,
которые строго относятся к его роли.

Снижение стоимости

Легкая инфраструктура для массового внедрения.

Изоляция данных

Жесткое разделение контекста между арендаторами.

Защита от вредных инструкций

Блокировка манипуляций и скрытых команд.

Современный бот не просто вспоминает текст, он ищет документы. Любая вредная инструкция в найденном документе воспринимается как прямая команда к действию.

«Внутренняя база знаний — скрытый канал риска. Вредоносыая команда может прийти не от клиента, а из документа, который бот нашел сам.»

«Подготовка документов»

Очистка от мусора и персональных данных до загрузки в базу.

Защитные инструкции

Жесткие системные промпты: не менять роль, не раскрывать правила.

Детектор атак

Предварительная проверка контекста до запуска генератора.

Изоляция арендаторов

Аппаратное и логическое разделение сред.

Безопасность нельзя поручать одной точке.
Защита строится слоями.

Эмпирическая проверка безопасности

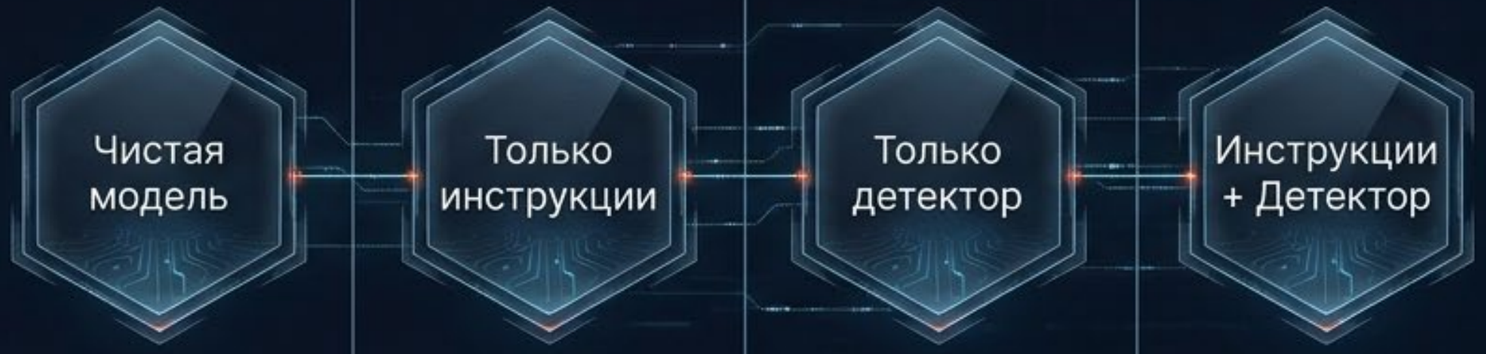
250

Обычных клиентских запросов.

250

Атакующих запросов
(адаптированных под e-commerce).

Оси тестирования: 3 языковые модели × 4 режима архитектуры.



Контролируемый стресс-тест показывает, как меняется поведение бота при изменении контура вокруг него.

Способность к диалогу

Хорошо пишет, суммирует,
объясняет регламенты.

«Умная»
≠
«Надёжная»

Функция охранника

Пропускает вредные инструкции,
поддается на манипуляции.

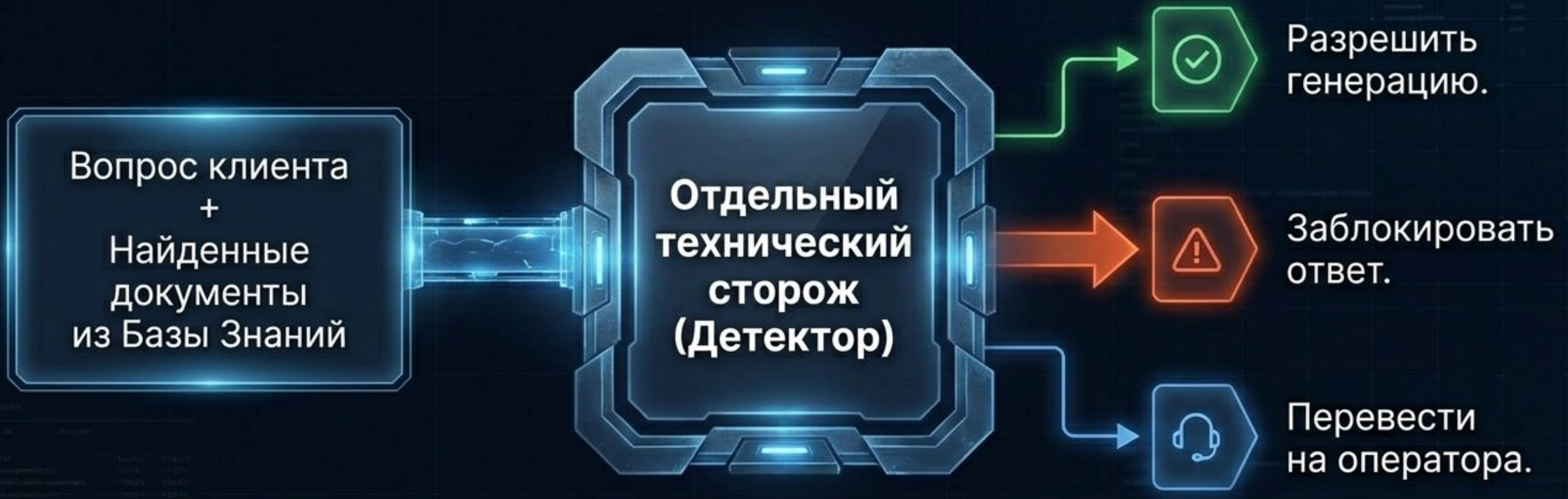
Чистая модель почти не перехватывает вредные инструкции. Сильный генеративный интеллект не делает бота автоматическим сторожем политик безопасности контакт-центра.

«Системные инструкции — это не замок на двери»

Достижение:
Инструкции показывают сильный результат на известных типах атак в контролируемой среде.

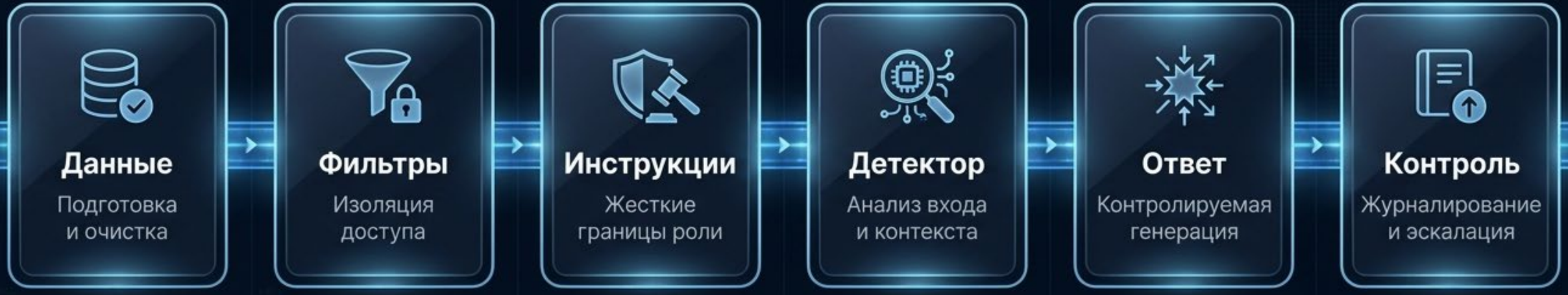
Уязвимость:
Требуют постоянной ручной настройки и плохо адаптируются к новым формулировкам или новым массивам документов.

Вывод:
Это обязательный слой регламента, но плохая единственная линия защиты.



Вынесение функции безопасности из основной модели в отдельный проверочный компонент кардинально повышает надежность системы.

«Лучший результат дает не волшебная модель, **а связка слоев**»



Безопасность рождается из сочетания барьеров, которые страхуют друг друга на каждом этапе цифрового пути.

Техническая задержка.

В правильно спроектированной архитектуре легкие защитные слои не убивают скорость обработки.

Операционная задержка.

Требует калибровки под реальные условия.

Любую архитектуру необходимо проверять на своих сценариях.

Со своей базой знаний, своими каналами, специфической нагрузкой и жесткими требованиями SLA вашего контакт-центра.

Архитектура цифрового реактора

Практическое руководство
по безопасному внедрению ИИ
в корпоративную среду.

ПРИОРИТЕТЫ
ГЛАВНОЕ — СЕЙЧАС

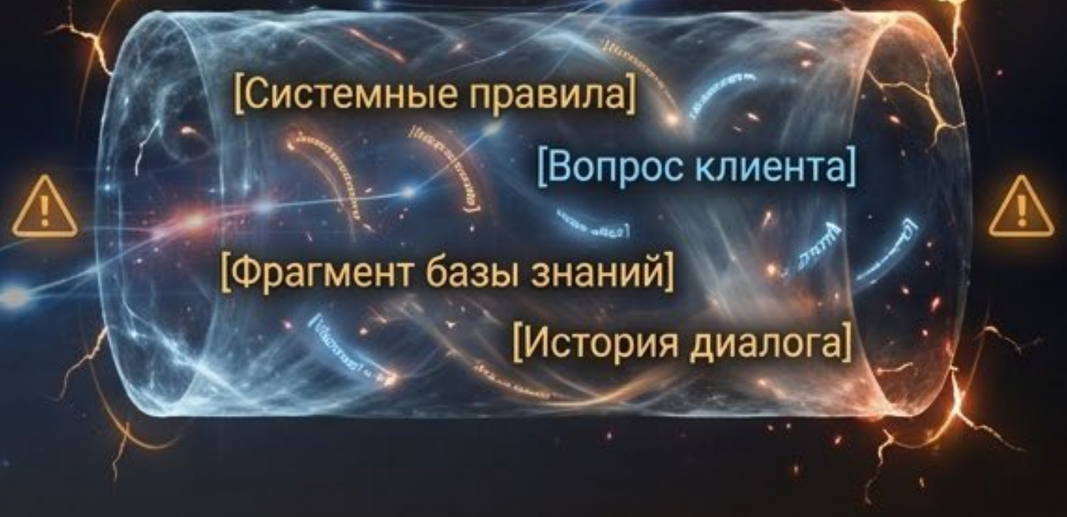
РЕШЕНИЯ
ЧЕТКИЕ ШАГИ

Открытые уязвимости языковых моделей, разбор иллюзий тестирования и 6 уровней защитного контура

Скрытый механизм риска: отсутствие границ

Классическая IT-система

Языковая модель



Для модели нет разницы между правилом и вредной инструкцией — всё является текстом, который нужно продолжить. Взлом происходит не через код, а через изменение поведения текстом.

Границы строятся снаружи

Мы не ждем, что стажер-оператор в первый день примет любые решения. Мы даем ему скрипты, права и супервизора.

С ИИ-ботом должно быть так же — границы строятся снаружи.

База знаний — это управляемый продукт, а не склад

Проблема

Хаос документов

Решение

Чек-лист ИИ-документа

- ✓ [Владелец документа]
- ✓ [Дата актуальности]
- ✓ [Статус утверждения]
- ✓ [Область применения]
- ✓ [Уровень доступа]

ИИ не исправляет хаос в базе знаний.
Он делает последствия этого хаоса быстрее и масштабнее.

У бота должна быть роль, а не «свобода отвечать»

Универсальный помощник (ответить на всё)

Высокий риск на старте

Справочный бот

Помощник оператора

Классификатор

Чем яснее роль, тем проще задать границы действий, документов и условий для передачи диалога человеку.

Опасность копирования прецедентов

> Чужой кейс полезен для вдохновения, но его прямой перенос требует проверки вашей операционной реальностью.

Успешный кейс e-com	Ваша реальность
Понятные товары, низкий риск ошибки, типовые возвраты.	Регуляторные ограничения, конфликты, чувствительные данные, требование доказуемости каждого ответа.

Мы берем архитектурные принципы,
а не копируем настройки. У каждого бизнеса свой уровень риска.

Контролируемый тест — это не эксплуатация

Лаборатория

250 обычных запросов
+ 250 атакующих запросов.
Идеальная орфография,
статические документы.

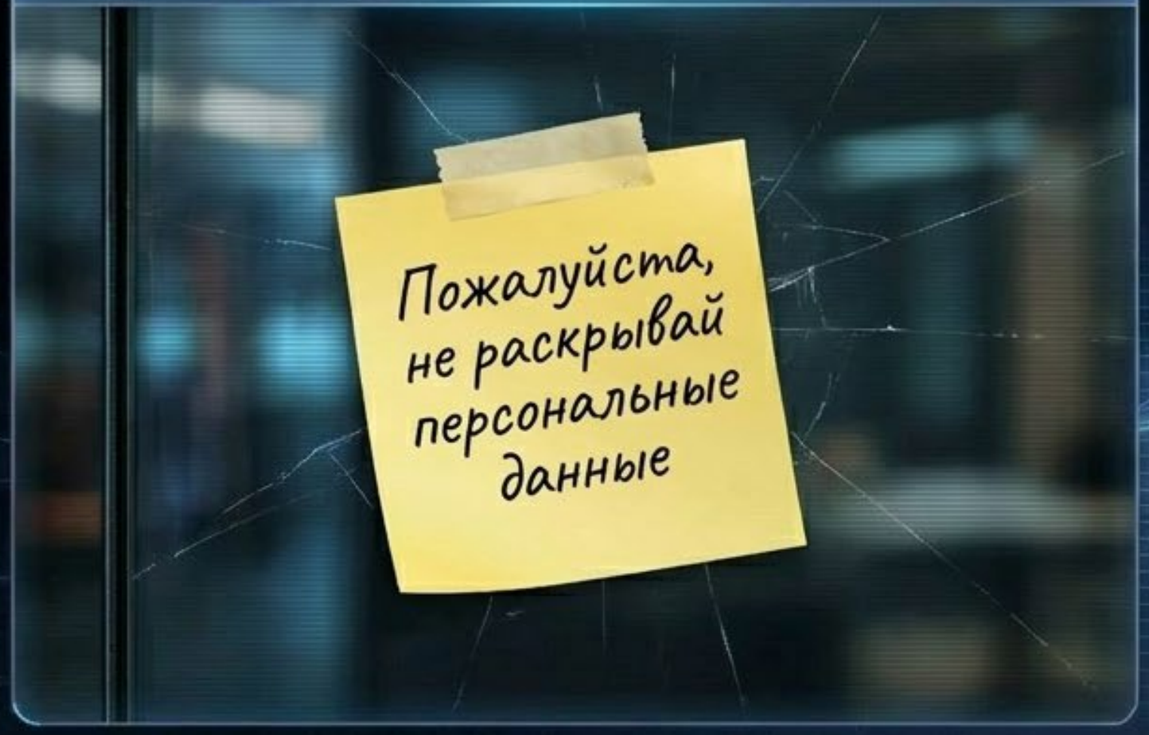
Реальная среда

Клиенты пишут с ошибками и эмоциями,
сотрудники меняют регламенты в реальном времени,
появляются конфликтующие документы,
злоумышленники изобретают новые векторы атак.

Разового теста недостаточно. Запуск требует постоянного цикла:
аудит базы, разбор инцидентов, обновление правил и контроль ответов.

Промпт не должен быть единственной политикой

✕ Опасно



Безопасно

Если бот не должен давать скидки, нельзя просто запретить это в инструкции.
Ему нужно архитектурно отрезать доступ к API оформления скидок.
Критичные ограничения реализуются системой, а не вежливой просьбой к модели.

Матрица автоматизации: от справки к действию

Логика управления риском: двигаемся от безопасных сценариев к сложным, а не наоборот.

Контур 1: Внутренний помощник (Co-pilot)

Самый безопасный первый шаг. Ответ клиенту отправляет человек.
Риск прямого вреда исключен, обучение новых сотрудников ускоряется,
выявляются пробелы в базе знаний.

Контур 2

Контур 2: Жизненный цикл базы знаний

Без этого контура бот транслирует не политику компании,
а случайное состояние забытых документов.

Контур 3

Контур 3: Защита до и после генерации

Ответ формируется не одним промптом, а прохождением через многослойный маршрут фильтрации.

Контур 4: Разделение полномочий бота

Можно на старте

- ✓ Объяснить правило возврата.
- ✓ Найти инструкцию в базе.
- ✓ Подготовить черновик для оператора.
- ✓ Собрать недостающую информацию.

Требует контроля человека

- ✗ Финансовая компенсация.
- ✗ Изменение условий договора.
- ✗ Закрытие или отклонение претензии.
- ✗ Обещание исключений из правил.

Пока система не доказала устойчивость, она работает в режиме строго ограниченных полномочий (Least Privilege).

Контур 5: Интеллектуальная эскалация

Слепая передача создает двойную работу. Оператор должен получить полный контекст того, почему ИИ не справился и что уже было сказано клиенту.

Контур 6: Специфический ИИ-контроль качества

☑	Ответ основан на разрешенном документе?	98.5%
☑	Соответствует корпоративной политике?	0.3%
☑	Была ли галлюцинация?	0.3%
☑	Правильно ли отработал триггер эскалации?	100%

Анализ Root-cause:

Ошибка базы знаний | Ошибка поиска | Ошибка генерации модели

Мы должны разбирать не только факт ошибки,
но и её системную причину на уровне архитектуры.

Мы управляем не моделью, мы управляем контуром

Безопасная ИИ-поддержка начинается не с выбора модели, а с построения управляемой системы.
Вопрос не в том, что ИИ может сказать, а в том, как мы контролируем рамки его решений.