для тех кто с тиктока

HERMES LEDGER / 57 МАТЕРИАЛОВ

Инструкции, ссылки, команды и промпты к роликам в одном документе. Найди знакомое название в оглавлении и открой нужный материал.

Редакция 12.09.2026. Ранние гайды дополнены пояснениями; похожие ролики объединены и подписаны.

Читать онлайн и скачивать приложения

Шаблоны, команды и учебные файлы одним ZIP

Оглавление

МАТЕРИАЛ 01

ИИ умеет собирать видео. Вот как получить MP4 через Remotion

Берём конкретную задачу: вертикальный ролик на 15 секунд с тремя сценами. Агент создаёт React-композицию, а Remotion превращает её в видео. Начни с текста и простых фигур - так легче увидеть проблемы с ритмом и читаемостью.

Установи Node.js и выполни команды по очереди:

```
npx create-video@latest --yes --blank my-video
cd my-video
npm install
npx remotion skills add
npm run dev
```

Открой папку my-video в редакторе с агентом. Последняя команда запускает Studio; точный локальный адрес появится в терминале. Команды соответствуют инструкции Remotion.

Задание для агента:

> Создай в этом Remotion-проекте композицию Promo: 1080×1920, 30 fps, ровно 450 кадров. Используй три сцены: 0-4 секунды - «Один понятный хук», 4-11 секунд - «Покажи конкретный результат», 11-15 секунд - «ТГК Hermes Ledger • @ledgerhermes». Чёрный фон, белая типографика, один лаймовый акцент. Без фотографий и музыки. Текст должен помещаться в безопасную область: минимум 100 px по бокам, 180 px сверху и 260 px снизу. Используй анимацию на основе текущего кадра Remotion, а не CSS-анимации по реальному времени. Выноси все тексты в один объект данных. Сделай короткие переходы между сценами, проверь кадры с самым длинным текстом и дай команду экспорта композиции Promo в MP4.

Посмотри ролик целиком без остановок. Если фразу не успеваешь прочитать, сократи её или увеличь длительность сцены. Затем в Studio нажми Render, выбери MP4/H.264 и подтверди экспорт. Это предусмотрено инструкцией по рендеру.

Чтобы сделать следующий ролик, сначала замени содержание трёх сцен и проверь тайминг. Не рассчитывай, что замена текста любой длины сохранит удачную композицию. Перед рабочим использованием ознакомься с условиями лицензии Remotion.

МАТЕРИАЛ 02

Нейросеть, которая отвечает даже без интернета

Установи Ollama, затем открой терминал:

```
ollama run qwen3:4b
```

Дождись окончания загрузки модели. Введи короткий вопрос, получи ответ, отключи интернет и задай новый вопрос. В этом примере используется локальный тег qwen3:4b, без облачного варианта.

Если не хватает памяти или ответ идёт слишком медленно, выйди из чата через /bye и попробуй меньшую модель. Для её первой загрузки снова понадобится интернет:

```
ollama run qwen3:1.7b
```

Во втором терминале посмотри, где считается ответ:

```
ollama ps
```

Поле PROCESSOR показывает размещение модели на CPU/GPU. Больший контекст требует дополнительной памяти; размер файла модели не равен всей памяти, необходимой для работы. Объяснение есть в FAQ Ollama.

Первый запрос из приложения, запущенного на том же компьютере. В современном Node.js с поддержкой fetch:

```
async function main() {
  const response = await fetch("http://localhost:11434/api/chat", {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({
      model: "qwen3:4b",
      stream: false,
      messages: [
        {
          role: "user",
          content: "Объясни замыкание в JavaScript на коротком примере."
        }
      ]
    })
  });

  if (!response.ok) {
    throw new Error("Ollama HTTP " + response.status);
  }

  const data = await response.json();
  console.log(data.message.content);
}

main().catch(console.error);
```

Это обращение к локальному API Ollama. Ollama должна быть запущена. На удалённом сервере localhost будет означать сам сервер, поэтому такой адрес не подключает опубликованный сайт к твоему ноутбуку автоматически.

Для принудительного локального режима можно задать OLLAMA_NO_CLOUD=1 и перезапустить Ollama. Это отключает облачные функции; способ установки переменной зависит от системы. Качество ответа всё равно нужно проверять - локальный запуск не делает модель безошибочной.

МАТЕРИАЛ 03

Как превратить сайт в приложение с иконкой на телефоне

Результат: иконка на домашнем экране и запуск в отдельном окне. Возможности и способ установки зависят от браузера и операционной системы.

Первое задание для агента:

> Изучи стек веб-проекта и добавь минимальную установку как PWA. Создай или обнови web app manifest, подключи его к странице и проверь корректные name, short_name, start_url, scope и display: standalone. Подготовь настоящие PNG-иконки 192×192 и 512×512 и apple-touch-icon. Учти базовый путь приложения, если оно размещено не в корне домена. Проверь MIME-тип manifest, отсутствие 404 у иконок и работу сайта по HTTPS. Не обещай автоматическое появление кнопки установки во всех браузерах. Пока не добавляй офлайн-кэширование. Дай порядок ручной проверки на Android и iPhone.

Manifest сообщает браузеру название, оформление и поведение приложения. HTTPS требуется для рабочего сайта; для локальной разработки есть исключение localhost. Состав manifest и требования описаны в руководстве MDN.

Проверка установки:

1. В Chrome DevTools открой Application → Manifest. Исправь ошибки manifest и загрузки иконок.
2. На Android открой сайт в Chrome. Используй предложенную установку или соответствующий пункт меню, если он доступен.
3. На iPhone открой сайт в Safari → «Поделиться» → «На экран Домой». В зависимости от версии iOS может быть дополнительная настройка открытия как веб-приложения.
4. Запусти именно новую иконку. Проверь начальную страницу, переходы между экранами и повторный запуск.

Платформенные различия разобраны в инструкции по установке веб-приложений.

Если нужен запуск без сети, дай отдельное задание:

> Добавь service worker с ограниченным кэшем оболочки приложения и понятной страницей офлайн. Не кэшируй персональные ответы API и авторизованные запросы по умолчанию. Предусмотри смену версии кэша, удаление старого кэша и понятное обновление приложения. Проверь первый запуск без ранее загруженного кэша, повторный запуск без сети и получение новой версии после возвращения интернета.

Офлайн-режим обеспечивается отдельно через service worker и кэш. Проверь, какие функции действительно доступны без сети, и показывай пользователю понятный статус.

МАТЕРИАЛ 04

Замени домен GitHub - и задай вопросы к коду проекта

Для публичного репозитория замени домен github.com на deepwiki.com. Оставь автора и название проекта, например открой корневую ссылку вида deepwiki.com/owner/repo, без хвоста /tree/ветка/папка. Это способ перехода, который показывает Cognition.

Если готового разбора нет, воспользуйся предложенной на сайте индексацией и дождись её завершения. Затем задавай вопросы последовательно:

1. «Где начинается запуск этого проекта? Покажи точку входа и связанные файлы. Для каждого вывода дай ссылку на исходник».
2. «Как проходит сценарий [конкретное действие пользователя]? Назови файлы по порядку: от интерфейса или входящего запроса до результата».
3. «Какие настройки и внешние сервисы обязательны для локального запуска? Подтверди ответ кодом и конфигурацией проекта».
4. «Где меняется [конкретная функция]? Объясни назначение ближайших зависимостей и какие другие сценарии может затронуть правка».
5. «Какие тесты проверяют эту функцию и что они утверждают? Сошлись на существующие файлы. Не придумывай отсутствующие проверки».

DeepWiki поддерживает вопросы по кодовой базе и ссылки на исходники; это описано в документации Cognition.

Мой предлагаемый порядок проверки: открыть упомянутые файлы, сверить ветку или коммит, найти вызывающий код и только потом переносить вывод в задачу агенту. Если ссылка ведёт на старый снимок проекта, проверь соответствующий участок в актуальной версии.

Задание своему агенту после разбора:

> Вот моя задача и предполагаемый маршрут по файлам: [вставь вывод]. Сначала сверь его с текущим checkout. Укажи несовпадения и найди реальные точки изменения. Затем предложи минимальную правку и способ проверить пользовательский результат.

Так ты приходишь к агенту с конкретным вопросом и можешь проверить, на чём основан ответ.

МАТЕРИАЛ 05

Запиши клики в браузере и получи готовый сценарий

Сначала установи Node.js. В папке проекта запусти:

```
npm init playwright@latest
```

При настройке согласись установить браузеры. Команда добавляет Playwright в проект по официальной инструкции.

Для первого опыта используй учебный сайт:

```
npx playwright codegen https://demo.playwright.dev/todomvc
```

В открывшемся браузере добавь задачу «Купить молоко» и отметь её выполненной. Inspector покажет код действий. Перед завершением записи выбери проверку видимого текста или значения и укажи результат, который должен сохраняться при повторном запуске. Codegen поддерживает действия и проверки - см. документацию.

Скопируй полный сгенерированный тест в tests/todo.spec.ts. В начале должны остаться импорты из @playwright/test и оболочка test(...), а не только несколько строк с кликами.

Запусти сохранённый сценарий:

```
npx playwright test tests/todo.spec.ts --headed
```

Убедись, что он проходит без твоей помощи и действительно проверяет ожидаемый текст/значение. Для своего проекта сначала запусти его обычным способом, затем передай его адрес в codegen.

Задание агенту для доработки записи:

> Сохрани пользовательский сценарий этого теста. Проверь локаторы и ожидаемый результат, добавь только недостающие проверки. Убери случайные ожидания фиксированного времени, если их можно заменить ожиданием состояния страницы. Сделай тест независимым от результатов предыдущего запуска и объясни, какие тестовые данные он создаёт. Не используй настоящие платежи и рабочие учётные данные.

Перед публикацией записи очисти её от введённых паролей и личных данных. Для ролика удобнее использовать учебный сайт или отдельный тестовый аккаунт.

МАТЕРИАЛ 06

Как превратить страницу сайта в данные для приложения

Начни с одной открытой страницы, которую разрешено обрабатывать. В Playground Firecrawl выбери Scrape, укажи URL и JSON. Цель первой попытки - проверить качество извлечения, а не собрать весь сайт.

Задание на извлечение:

> Верни только карточки каталога, действительно присутствующие на этой странице. Для каждой карточки извлеки название, цену в точности как она написана и абсолютную ссылку. Если значение отсутствует, верни null. Не придумывай карточки, валюту и скидки. Результат должен соответствовать схеме.

Схема полей:

```
{
  "type": "object",
  "properties": {
    "items": {
      "type": "array",
      "items": {
        "type": "object",
        "properties": {
          "title": { "type": ["string", "null"] },
          "price": { "type": ["string", "null"] },
          "url": { "type": ["string", "null"] }
        },
        "required": ["title", "price", "url"],
        "additionalProperties": false
      }
    }
  },
  "required": ["items"],
  "additionalProperties": false
}
```

В API v2 схема передаётся внутри объекта formats, как показано в документации JSON-режима. Ниже самостоятельный пример для современного Node.js. Сохрани его как extract.mjs, задай FIRECRAWL_API_KEY и SOURCE_URL в окружении сервера или локального терминала и запусти node extract.mjs. Не размещай ключ в клиентской странице.

```
const apiKey = process.env.FIRECRAWL_API_KEY;
const sourceUrl = process.env.SOURCE_URL;

if (!apiKey || !sourceUrl) {
  throw new Error("Задай FIRECRAWL_API_KEY и SOURCE_URL");
}

const schema = {
  type: "object",
  properties: {
    items: {
      type: "array",
      items: {
        type: "object",
        properties: {
          title: { type: ["string", "null"] },
          price: { type: ["string", "null"] },
```

```
          url: { type: ["string", "null"] }
        },
        required: ["title", "price", "url"],
        additionalProperties: false
      }
    }
  },
  required: ["items"],
  additionalProperties: false
};

const response = await fetch("https://api.firecrawl.dev/v2/scrape", {
  method: "POST",
  headers: {
    "Authorization": "Bearer " + apiKey,
    "Content-Type": "application/json"
  },
  body: JSON.stringify({
    url: sourceUrl,
    formats: [{
      type: "json",
      prompt: "Извлеки только карточки со страницы: title, price, абсолютный url. Отсутствующее значение - nu
ll.",
      schema
    }]
  })
});

if (!response.ok) {
  throw new Error("Firecrawl HTTP " + response.status);
}

const result = await response.json();
if (!result.success || !Array.isArray(result.data?.json?.items)) {
  throw new Error("Не удалось получить массив items");
}

console.log(JSON.stringify(result.data.json, null, 2));
```

Проверь несколько карточек вручную: название, цена, валюта, ссылка и отсутствие дублей. Структура JSON не гарантирует точность содержимого. Для нескольких страниц используй ограниченный обход со списком разрешённых адресов; различия между scrape и crawl описаны здесь. У облачного сервиса есть расходуемые лимиты, поэтому сначала проверь малый объём.

МАТЕРИАЛ 07

Заявка из формы сразу приходит в Telegram: собираем в n8n

Соберём уведомление для владельца формы. Понадобится запущенный n8n, новый Telegram-бот и чат, в который ты хочешь получать заявки.

В n8n добавь Form Trigger. Создай три поля: «Имя», «Контакт», «Вопрос». Сделай контакт и вопрос обязательными. Открой Test URL, чтобы отправить пробные данные. У формы есть отдельные тестовый и рабочий адреса - описание Form Trigger.

В Telegram открой проверенный аккаунт BotFather, отправь /newbot и создай бота. Полученный токен сохрани в Telegram Credentials внутри n8n. Не размещай его в форме или публичном коде. Этот способ описан в настройке Telegram credentials.

Как получить свой chat ID без стороннего бота:

1. В отдельном временном workflow добавь Telegram Trigger с credentials нового бота.
2. Включи ожидание тестового события и отправь своему боту /start.
3. В выходном JSON найди message → chat → id и скопируй число целиком.
4. Останови тест и удали временный workflow. В основном сценарии оставь запуск от формы.

Поле message.chat.id используется в исходнике Telegram Trigger. Для получения событий n8n должен иметь доступный Telegram адрес вебхука; у локального инстанса без внешнего адреса этот тест не сработает.

Вернись к форме. Подключи Telegram, выбери Resource: Message и Operation: Send Message. В Chat ID вставь число своего чата. В Text включи выражения или перетащи поля из результата формы:

```
Новая заявка

Имя: {{ $json["Имя"] }}
Контакт: {{ $json["Контакт"] }}
Вопрос: {{ $json["Вопрос"] }}
```

Названия в квадратных скобках должны точно совпадать с ключами, которые вернул Form Trigger. Для начала оставь обычный текст без Markdown/HTML-разметки. Настройки отправки есть в документации Telegram node.

Отправь одну заявку через Test URL и проверь все поля сообщения. Затем опубликуй workflow и размести Production URL. Проверь ещё одну заявку уже через рабочий адрес. n8n должен продолжать работать, чтобы принимать новые формы и отправлять уведомления.

МАТЕРИАЛ 08

Одна картинка может тормозить весь сайт. Вот как её облегчить

Открой Squoosh: изображение обрабатывается локально, а ползунок помогает сравнить исходник и результат.

Моя стартовая памятка:

Что сжимаем	С чего начать	Что проверить
Фото	Подобрать размеры под отображение, сравнить WebP и AVIF	Лица, волосы, небо, мелкие детали
Скриншот	Сначала уменьшить лишние размеры; сравнить PNG и WebP lossless	Читаемость мелкого текста при 100%
Растровый логотип	Сохранить прозрачность; начать с PNG или WebP lossless	Контуры на светлом и тёмном фоне

Это порядок подбора, а не универсальные значения качества. Для экрана с высокой плотностью пикселей может понадобиться версия крупнее, чем размер блока в CSS. Если логотип уже в SVG, сначала проверь возможность оставить вектор.

Сохраняй исходник отдельно. Запиши размер до и после, затем открой страницу с новым файлом. Большая разница в килобайтах сама по себе не доказывает такую же разницу во времени загрузки.

Промпт для Codex:

«Найди растровые изображения в проекте и покажи 15 самых тяжёлых: путь, размер файла, размеры в пикселях и где используется. Сопоставь их с отображаемыми размерами, если можешь это проверить. Предложи варианты оптимизации с учётом мобильного экрана, прозрачности и плотности пикселей. Сначала сделай новые версии для сравнения, сохрани оригиналы. После выбора обнови ссылки и проверь, что изображения загружаются».

Форматы и локальную обработку описывает репозиторий Squoosh.

Файлы к ролику

- GUIDE.md
- PROMPT.txt

Скачать все файлы этого гайда.

МАТЕРИАЛ 09

Из видео в субтитры: команда для Whisper

Локальный Whisper распознаёт речь; для запуска нужны Python и FFmpeg. Инструкции проекта и список моделей - в официальном репозитории.

В приложенном GUIDE.md - отдельные шаги для Windows и macOS, создание окружения и проверка установки.

Установи пакет в активированном окружении:

```
python -m pip install -U openai-whisper
```

Перейди в папку с роликом. Эта команда целиком занимает одну строку:

```
whisper video.mp4 --model small --language Russian --output_format srt
```

Для пути с пробелами используй кавычки. При первом запуске потребуется скачать модель. Вариант small здесь выбран как стартовый для русской речи, а не как самый точный для любой записи.

Формат SRT содержит пронумерованные фрагменты текста с временем начала и конца. Мини-пример лежит в example.srt. Параметры вывода доступны в исходном коде CLI.

После импорта в редактор проверь имена, цифры, паузы и таймкоды. У Whisper бывают ошибки и выдуманные фразы, особенно на сложном аудио; это описано в model card.

Промпт для обработки папки:

«Напиши локальный скрипт для запуска установленного Whisper над видео в выбранной папке. Сохраняй SRT в отдельную папку и не меняй оригиналы. Обрабатывай файлы последовательно. Поддержи пробелы в путях, пропуск готовых результатов и журнал ошибок. Проверяй существование непустого SRT после каждого запуска. Покажи команду запуска для моей ОС. Сначала обработай один короткий файл».

Сначала проверь качество на небольшом фрагменте своей записи - так проще выбрать модель.

Whisper: установка и запуск

Ниже локальная установка официального openai-whisper. Ориентир для окружения - Python 3.11, входящий в диапазон совместимости, указанный в README проекта. Первому запуску нужен интернет для скачивания модели; распознавание затем выполняется на твоём компьютере.

Windows

1. Установи Python 3.11 и pip с python.org. Проверь в новом терминале: `py -3.11 --version`.
2. Установи FFmpeg. Если уже есть Chocolatey, выполни `choco install ffmpeg`. Другой путь: возьми сборку по ссылкам со страницы FFmpeg, распакуй её и добавь каталог `bin` в

пользовательский PATH.

3. Открой новый терминал и проверь: `ffmpeg -version`.
4. В папке для работы выполни команды. Активация окружения не требуется - указан полный путь к программе внутри него.

```
py -3.11 -m venv .venv
.venv\Scripts\python.exe -m pip install -U openai-whisper
.venv\Scripts\whisper.exe "video.mp4" --model small --language Russian --output_format srt --output_dir subti
tles
```

В командах выше обратные слеши обозначают обычные разделители пути Windows. После распаковки используй команды из файла command_windows.txt, где они записаны готовыми для вставки.

macOS

1. Установи Python 3.11 с python.org, если его ещё нет.
2. При установленном Homebrew запусти `brew install ffmpeg`, затем `ffmpeg -version`.
3. В рабочей папке выполни:

```
python3.11 -m venv .venv
.venv/bin/python -m pip install -U openai-whisper
.venv/bin/whisper "video.mp4" --model small --language Russian --output_format srt --output_dir subtitles
```

Заменяй video.mp4 на свой путь. Папка subtitles появится автоматически. Скорость зависит от модели, процессора или видеокарты и длины записи. Если нет ускорителя, распознавание может занять заметное время.

После запуска

Проверь, что subtitles/video.srt появился и не пустой. Импортируй его в редактор с поддержкой SRT. Оформление и переносы строк подбирай в редакторе, затем сверь распознанную речь с оригиналом.

example.srt - вручную написанный учебный пример формата, не результат обработки реального видео. PROMPT.txt - задание для создания скрипта обработки папки.

Параметры вывода: Whisper CLI. Ограничения распознавания: model card.

Файлы к ролику

- GUIDE.md
- PROMPT.txt
- command_macos.txt
- command_windows.txt
- example.srt

Скачать все файлы этого гайда.

МАТЕРИАЛ 10

Большой CSV можно разобрать без загрузки в нейросеть

DuckDB читает CSV непосредственно в запросе и обычно определяет формат автоматически. Если это не сработало, разделитель, типы и заголовок можно задать явно. Документация CSV.

Скачай DuckDB CLI для своей ОС, распакуй его и открой в папке с sales.csv. В Windows исполняемый файл может называться duckdb.exe; в терминале из той же папки запускай .\duckdb.exe. В macOS/Linux - ./duckdb, либо duckdb, если программа добавлена в PATH.

В приложении - учебный sales.csv с вымышленными строками и пять отдельных SQL-файлов:

1. 01_totals.sql - количество строк и сумма.
2. 02_top_categories.sql - категории по сумме.
3. 03_missing.sql - строки с пропусками.
4. 04_duplicates.sql - повторяющиеся order_id.
5. 05_export.sql - сводка в новом CSV.

В консоли сначала посмотри данные:

```
SELECT * FROM 'sales.csv' LIMIT 5;
DESCRIBE SELECT * FROM 'sales.csv';
```

Затем, например:

```
SELECT category, SUM(amount) AS total
FROM 'sales.csv'
GROUP BY category
ORDER BY total DESC;
```

Это сумма всех строк, включая дубли. Учебный файл специально содержит повтор и пропуск: суммирование само по себе не очищает данные. Ожидаемые результаты указаны в GUIDE.md. Чтение CSV, экспорт результата.

Промпт:

«Вот структура CSV и несколько вымышленных строк: [ПРИМЕР]. Напиши запрос DuckDB для [ВОПРОС]. Укажи допущения о типах, пропусках и дублях. Не утверждай, что посчитал мой файл. Добавь небольшой контрольный запрос для сверки результата».

Не подставляй выдуманные ответы вместо запуска SQL: вычисление делает DuckDB на твоём файле.

Учебный CSV и пять запросов

Все данные вымышлены. Файл содержит 8 строк, 7 разных order_id, один повтор order_id=1005 и один пропуск amount.

Запусти DuckDB CLI из папки с учебными файлами. Сначала посмотри первые строки: `SELECT * FROM 'sales.csv' LIMIT 5;`. В интерактивной консоли выполняй нужный файл командой `.read 01_totals.sql`, заменяя имя для остальных запросов.

Ожидаемые результаты:

Файл	Результат
01_totals.sql	row_count=8, unique_orders=7, total=595
02_top_categories.sql	Tools=280; Courses=250; Books=65
03_missing.sql	Строка order_id=1006, amount=NULL
04_duplicates.sql	order_id=1005, copies=2
05_export.sql	Новый summary.csv с тремя категориями

Суммы здесь включают обе строки order_id=1005. SUM пропускает NULL, но не удаляет дубли. Правило очистки зависит от смысла данных. Запрос 05_export.sql создаёт summary.csv и может перезаписать файл с таким именем: используй учебную папку и выбери другое имя, если оно уже занято.

На своём файле проверь разделитель, заголовок и типы столбцов. Команда: `DESCRIBE SELECT * FROM 'sales.csv';`.

DuckDB CLI, CSV, экспорт.

Файлы к ролику

- 01_totals.sql
- 02_top_categories.sql
- 03_missing.sql
- 04_duplicates.sql
- 05_export.sql
- GUIDE.md
- PROMPT.txt
- sales.csv

Скачать все файлы этого гайда.

МАТЕРИАЛ 11

Почему люди закрывают твой сайт? Посмотри запись сессии

PostHog Session Replay воспроизводит взаимодействия с интерфейсом. Установку под свой стек бери из официальной инструкции.

Сначала запиши собственный тестовый путь: зайди на страницу, прокрути её, нажми кнопку, заполни форму вымышленными данными. Найди запись в Replay и проверь, что важные взаимодействия видны. Как смотреть записи.

Семь вопросов для разбора:

1. Понятно ли по первому экрану, что делать дальше?
2. Прокручивают ли до основного действия?
3. Кликают ли по тому, что выглядит кнопкой, но ею не является?
4. После нажатия видно ли, что действие началось?
5. На каком поле формы чаще останавливаются?
6. Возвращаются ли назад после конкретного шага?
7. Повторяется ли проблема в нескольких сессиях на похожих устройствах?

Записывай наблюдение отдельно от объяснения. «Три раза нажал кнопку» - наблюдение. «Не заметил загрузку» - гипотеза, которую ещё нужно проверить.

Начальная конфигурация маскирования для веб-SDK, внутри posthog.init:

```
session_recording: {
  maskAllInputs: true,
  maskTextSelector: '*'
}
```

Она скрывает ввод и текст, но не является полной настройкой приватности: URL, изображения, содержимое сторонних элементов и дополнительные виды записи проверяются отдельно. Отключи захват чувствительных экранов и согласуй запуск записи с выбранными в продукте настройками согласия. Примеры - в документации маскирования.

Промпт:

«Вот обезличенные наблюдения из сессий: [СПИСОК]. Отдели факты от гипотез. Для каждой повторяющейся проблемы предложи минимальное изменение интерфейса и способ проверить результат. Не делай выводы о мотивах человека и не обещай рост конверсии».

Тариф и объём записи выбирай по своей нагрузке; здесь нет обещания безлимитного хранения.

Файлы к ролику

- GUIDE.md
- PROMPT.txt

- privacy_options.js
- replay_review.csv

Скачать все файлы этого гайда.

МАТЕРИАЛ 12

Проверь нейросеть на своём примере ещё до установки

Hugging Face Spaces - каталог приложений и демо. Публичные Spaces позволяют открыть работающий интерфейс; размещение своей копии и доступные ресурсы имеют отдельные условия. Как устроены Spaces.

Три отправные точки:

1. BRIA RMBG 2.0 - попробуй удаление фона. Для проверки возьми предмет с тонкими краями или сложным окружением.
2. FLUX.1 Schnell - попробуй генерацию изображения. Дай один и тот же запрос несколько раз и сравни, насколько стабильно соблюдается композиция.
3. Qwen3 VL Demo - приложи картинку и задай вопрос по содержимому. Ответы о мелких деталях сверяй с самим изображением.

У приложения может появиться очередь, лимит или требование входа. Если оно остановлено, посмотри другие демо той же модели у её разработчика.

Шаблон сравнения лежит в comparison.csv. В нём есть задача, входной пример, ссылка, результат, замеченные ошибки, время ожидания, условия доступа и лицензия модели.

Мой порядок проверки: обычный пример → сложный пример → повтор обычного примера. Это помогает отличить удачный единичный результат от подходящего инструмента.

Промпт перед интеграцией:

«Вот задача, выбранная модель, её документация и результаты моих проб: [ССЫЛКИ И ЗАМЕТКИ]. Объясни, какие возможности подтверждены примерами. Проверь условия использования, вариант API или локального запуска и ограничения. Предложи минимальную интеграцию с обработкой ошибок. Не считай публичное демо гарантированно доступным API».

Файлы к ролику

- DEMO_LINKS.md
- PROMPT.txt
- comparison.csv

Скачать все файлы этого гайда.

МАТЕРИАЛ 13

ИИ опять сделал не тот экран? Покажи ему четыре блока

Excalidraw позволяет набросать экран из простых фигур и выгрузить изображение. Экспорт в PNG, SVG и редактируемый формат описан в репозитории проекта.

К посту приложены три файла:

- landing.excalidraw - первый экран, блоки пользы и основное действие.
- catalog.excalidraw - поиск, фильтры и карточки.
- dashboard.excalidraw - меню, сводка и список последних действий.

Это заготовки расположения блоков. Открой нужный файл в Excalidraw, замени подписи и переставь элементы под свой сценарий. Потом выдели нужную область и экспортируй PNG через меню экспорта изображения.

Мой способ работы: один экран → одна понятная задача пользователя → одна главная кнопка. Сначала структура, затем визуальные детали.

Промпт для реализации:

«Прикладываю набросок экрана. Стек: [СТЕК]. Назначение: [ЗАДАЧА]. Главное действие пользователя: [ДЕЙСТВИЕ]. Прочитай текущие компоненты проекта и реализуй этот экран, сохранив расположение и порядок блоков. Используй существующие шрифты, цвета и компоненты. Подписи на схеме обозначают функции блоков, а не обязательный финальный текст. Для телефона предложи порядок сворачивания меню и колонок. Добавь состояния загрузки, пустого результата и ошибки там, где они нужны. Проверь экран на ширине 390 и 1440 px. Покажи скриншоты и перечисли расхождения с наброском, которые потребовались для удобства интерфейса».

После первой версии сравни три вещи: порядок чтения, расположение главного действия и поведение на телефоне.

Редактируемую схему сохрани рядом с PNG: её проще поправить для следующего экрана. Возможности экспорта.

Файлы к ролику

- GUIDE.md
- PROMPT.txt
- catalog.excalidraw
- catalog_preview.svg
- dashboard.excalidraw
- dashboard_preview.svg
- landing.excalidraw
- landing_preview.svg

Скачать все файлы этого гайда.

МАТЕРИАЛ 14

Сайт упал, а ты узнаёшь последним. Настрой уведомление

Uptime Kuma поддерживает HTTP(s)-проверки и уведомления в Telegram. В приложении - compose.yaml для локальной проверки или установки на отдельном сервере с Docker Compose.

Из папки с файлом запусти:

```
docker compose up -d
```

Открой http://localhost:3001 и создай учётную запись администратора. В примере порт привязан к 127.0.0.1. Для удалённого сервера открой SSH-туннель:

```
ssh -L 3001:127.0.0.1:3001 user@SERVER_IP
```

Дальше открой localhost:3001 на своём компьютере. Закрытие туннеля закроет доступ к панели через него, но контейнер на сервере продолжит работать.

Добавь монитор HTTP(s): адрес сайта, понятное имя и интервал. Для начала можно выбрать 60 секунд и одну повторную попытку. Учти, что попытки и таймаут увеличивают задержку уведомления.

Для Telegram:

1. Создай бота через @BotFather и сохрани его токен.
2. Открой чат с ботом и нажми Start.
3. В настройках уведомления Kuma выбери Telegram и введи токен.
4. Получи Chat ID через кнопку автоматического получения в форме или укажи известный ID своего чата.
5. Отправь тест и привяжи уведомление к монитору.

Поля и получение Chat ID можно проверить в компоненте Telegram проекта.

Тест без остановки сайта: создай отдельный монитор с рабочим URL, дождись статуса UP, временно укажи в нём адрес https://example.invalid и дождись DOWN. Верни рабочий URL и проверь восстановление. Домен .invalid специально не предназначен для работающего сайта.

Для постоянной работы нужен отдельный работающий хост. Бесплатный исходный код не означает бесплатный сервер. HTTP-проверка показывает доступность ответа; для ключевого сценария приложения может понадобиться более содержательная проверка.

compose.yaml

```
services:
  uptime-kuma:
    image: louislam/uptime-kuma:2
    restart: unless-stopped
    ports:
      - "127.0.0.1:3001:3001"
```

```
    volumes:
      - uptime-kuma-data:/app/data

volumes:
  uptime-kuma-data:
```

Файлы к ролику

- GUIDE.md
- compose.yaml

Скачать все файлы этого гайда.

МАТЕРИАЛ 15

7 API, которые оживят твой MVP

Интерфейс готов, а внутри заглушки? Выбери один источник данных и сначала покажи на экране пять настоящих записей.

API	Что можно собрать	Что учесть
Open-Meteo	Погода по выбранному городу	Бесплатный открытый доступ имеет условия; коммерческое использование проверяй отдельно
NASA API	Космический календарь и фотографии	Для выбранного API проверь ключ и лимит; DEMO_KEY подходит для знакомства
Open Library	Поиск книг и книжный трекер	Не у каждой книги есть обложка и полные метаданные
Frankfurter	История валютных курсов	Справочные курсы не равны котировке, по которой банк проведёт обмен
REST Countries	Справочник стран или географический квиз	Проверь нужные поля и язык названий
TMDB	Киноафиша и списки просмотра	Нужны доступ к API и соблюдение требований к атрибуции
Unsplash	Поиск фотографий	Соблюдай правила API, показа изображений и указания автора

Порядок подключения: открой документацию → получи один ответ → опиши нужные поля → добавь загрузку, пустой результат и ошибку → проверь ограничение запросов.

Промпт:

Изучи проект и подключи [API] через отдельный адаптер. Реализуй только сценарий [СЦЕНАРИЙ]. Секретные ключи храни на сервере. Добавь типы ответа, loading, empty и error states, timeout, обработку лимита и небольшой кэш. При отсутствии поля показывай понятную заглушку. Подготовь mock для разработки и env.example без секретов. Покажи, как проверить один успешный и один неуспешный запрос.

МАТЕРИАЛ 16

Не начинай MVP с пустой папки

Стартер полезен, если совпадает с будущим продуктом. Сначала запусти исходную версию, затем убирай ненужные модули по одному.

Основа	Для какой задачи
Next.js SaaS Starter	SaaS с учётными записями, командами и Stripe
Open SaaS	Продукт с авторизацией, письмами, платежами и фоновыми задачами
Vercel Chatbot	AI-чат с историей диалогов
Next.js Commerce	Витрина интернет-магазина
Vercel Platforms	Приложение для нескольких клиентов с отдельными поддоменами
Next.js + Supabase	Пример связки Next.js и Supabase
Payload Website Template	Сайт с CMS и редактируемым контентом

Открытый исходный код не оплачивает базу, почту, AI-запросы и хостинг. Обязательные внешние сервисы и лицензию смотри в README выбранного репозитория.

Промпт:

> Моя идея: [ИДЕЯ]. Сравни эти стартеры по соответствию продукту, сложности запуска, базе, auth, платежам и лишним зависимостям. Выбери основной и запасной вариант. Для основного прочитай текущий README, выпиши команды запуска, обязательные env-переменные без значений и модули, которые можно убрать. Сначала добейся запуска исходного примера, затем добавляй мой сценарий.

МАТЕРИАЛ 17

Где вайбкодеру найти нормальный AI-хакатон

Это навигация по площадкам из роликов «5 хакатонов» и «Где найти AI-хакатон». Конкретные сроки, призы и правила смотри на странице выбранного события.

1. lablab.ai - AI-хакатоны и командные проекты.
2. Devpost - каталог с фильтрами по срокам и формату.
3. MLH Global Hack Week - тематические недели с заданиями и воркшопами.
4. DoraHacks - хакатоны и buildathons.
5. Encode Club - программы и технологические события.
6. Devfolio - онлайн- и офлайн-хакатоны.
7. ETHGlobal - события с фокусом на Ethereum; уместны, когда проект связан с этой экосистемой.

Перед выбором выпиши: дедлайн в своём часовом поясе, допустимые страны и возраст, командный размер, обязательный стек, правила ранее начатых проектов и формат демонстрации. Участие само по себе не означает приз.

Промпт для заявки:

> Вот правила события: [ССЫЛКА И ТЕКСТ]. Моя идея: [ИДЕЯ]. Упакуй её в заявку: проблема, конкретный пользователь, текущий способ решения, наш сценарий и отличие. Ограничь MVP тем, что можно показать за доступные [ЧАСЫ]. Раздели обязательные функции и всё, что откладываем. Напиши сценарий демо на 60 секунд. Не придумывай пользователей, выручку и результаты.

МАТЕРИАЛ 18

Какой backend выбрать для MVP

Основа	Когда смотреть	Важное различие
Supabase	Нужны Postgres, auth, storage и realtime	Права на данные и политики доступа настраиваются отдельно
Convex	Реактивный интерфейс и серверные функции	Приложение строится вокруг модели данных и функций Convex
Turso	Подходит SQLite-подход	Это не готовые auth, почта и хранение файлов
Firebase	Web/mobile в экосистеме Google	Сначала выбери конкретную базу и модель доступа
Neon	Нужен управляемый Postgres	Остальные части backend подбираются отдельно
Appwrite	Нужны backend-сервисы в одной платформе	Сравни облачный и самостоятельный запуск
PocketBase	Компактный прототип или внутренний инструмент	Проверь ограничения текущей версии и план резервирования

Не выбирай только по бесплатному старту. Сначала запиши связи данных, realtime, файлы, фоновые задачи и способ восстановления.

Промпт:

> Вот сценарии приложения и пример данных: [ОПИСАНИЕ]. Сравни перечисленные основы по отношениям между данными, auth, realtime, файлам, переносимости и ожидаемой нагрузке. Отдели возможности самой базы от дополнительных сервисов. Выбери один вариант и один запасной, назови главный компромисс каждого. Предложи минимальную схему и один пользовательский сценарий для проверки интеграции.

МАТЕРИАЛ 19

AI снова сделал серые карточки?

Библиотека	Что искать
Magic UI	Bento-блоки, анимированные акценты и фоны
React Bits	Эффекты текста, фоны и интерактивные React-компоненты
Aceternity UI	Hero-секции, карточки и эффекты наведения
shadcn/ui	Базовые компоненты, код которых живёт в проекте
Motion Primitives	Небольшие анимационные примитивы
daisyUI	Компоненты и темы для Tailwind
FlyonUI	Tailwind-компоненты и интерактивные блоки

Возьми одну основу и максимум один источник акцентов. Проверь совместимость версий, лицензию нужного компонента и его работу на телефоне.

Промпт:

> Встрой [КОМПОНЕНТ, ССЫЛКА] в экран [ЭКРАН]. Сохрани текущие шрифты, цвета, сетку и размеры элементов. Добавь только нужные зависимости. Проверь клавиатуру, мобильную ширину и prefers-reduced-motion. Сравни загрузку страницы до и после. Покажи, где отключить эффект, если он мешает содержанию.

МАТЕРИАЛ 20

Не проси AI писать авторизацию с нуля

Решение	Подходящий сценарий
Better Auth	Auth внутри TypeScript-приложения с контролем интеграции
Clerk	Готовые экраны входа и управление пользователями
WorkOS AuthKit	Организации и B2B-сценарии
Kinde	Управляемые пользователи и организации
Supabase Auth	Проект уже использует Supabase
Firebase Authentication	Проект уже использует Firebase
Auth0	Разные способы входа и интеграции

Вход подтверждает, кто пользователь. Право читать конкретную запись и выполнять действие всё равно проверяется сервером. Стоимость SSO, организаций и отдельных функций уточняй в текущем тарифе.

Промпт:

> Изучи стек и выбери auth-решение из списка. Нужны [СПОСОБЫ ВХОДА], [РОЛИ], [ОРГАНИЗАЦИИ ИЛИ ИХ ОТСУТСТВИЕ]. Используй официальный SDK и документацию. Опиши жизненный цикл сессии, восстановление доступа и серверные проверки прав. Добавь проверку сценариев: гость, свой аккаунт, чужая запись, завершённая сессия. Не реализуй собственную криптографию.

МАТЕРИАЛ 21

Локально работает - это ещё не релиз

Площадка	На что смотреть
Vercel	Next.js, предпросмотр изменений, функции
Cloudflare Workers	Web/API в среде Workers; совместимость runtime
Render	Web-сервисы, workers, cron, базы
Netlify	Frontend и функции с публикацией из Git
Deno Deploy	Поддерживаемые приложения и API в среде Deno
Koyeb	Контейнеры и web-сервисы
Railway	Backend, Docker и базы данных

Перед выбором выясни build-команду, runtime, порт, постоянный диск, базу, cron и фоновые задачи. Нельзя предполагать, что любой хостинг frontend запустит постоянный процесс.

Промпт:

> Изучи репозиторий. Выпиши runtime, build, команду запуска, порт, env-переменные без значений, потребность в диске, базе и фоновых задачах. Подбери площадку из списка по текущей документации. Подготовь конфигурацию, health check, миграции и способ отката. После релиза проверь основной сценарий через опубликованный адрес.

МАТЕРИАЛ 22

Пять API без ключа для первого проекта

Это отдельная ранняя подборка. В ней были погода, валюты, праздники, книги и игровые данные.

Источник	Первый сценарий
Open-Meteo	Погода по координатам
Frankfurter	Курсы валют на выбранную дату
Nager.Date	Календарь государственных праздников
Open Library	Поиск книги по названию
PokéAPI	Энциклопедия персонажей и способностей

Открываемый пример: данные Pikachu. Отсутствие ключа не отменяет условия использования, ограничения нагрузки и необходимость кэширования.

Промпт:

> Сделай один экран на данных [API]: поле поиска, результат, состояние загрузки и ошибка. Возьми только нужные поля. Не запрашивай API на каждый рендер и не скачивай весь каталог. Добавь задержку поиска, кэш и ссылку на источник. Покажи пример ответа и сопоставление полей с интерфейсом.

МАТЕРИАЛ 23

AGENTS.md: файл с правилами проекта

В корне репозитория создай AGENTS.md: краткие инструкции, которые относятся к работе с этим кодом. Codex поддерживает такой файл; особенности поиска вложенных инструкций описаны в официальной документации.

Шаблон для заполнения:

```
# Работа с проектом

## Команды
- Установка: [проверенная команда]
- Запуск: [проверенная команда]
- Тесты: [проверенная команда]
- Сборка: [проверенная команда]

## Правила
- Прочитай затрагиваемый код перед изменением.
- Сохраняй существующие соглашения и компоненты.
- Не удаляй чужие незавершённые изменения.
- Перед изменением архитектуры прочитай DECISIONS.md.

## Готовность
- Проверен пользовательский результат задачи.
- Запущены нужные проверки: ограничения названы.
- Для UI проверены 390 и 1440 px.
- В ответе перечислены изменения и способ проверки.
```

Замени скобки настоящими командами из проекта. Сам файл не запускает проверки и не гарантирует результат. Если агент повторяет ошибку, добавь короткое конкретное правило, которое помогает её избежать.

Скачать AGENTS.md.

МАТЕРИАЛ 24

Файл, который не даёт Codex забывать твои решения

DECISIONS.md хранит причины архитектурных решений. Для каждого запиши: что выбрали, почему и при каком условии пересматриваем.

```
# Решения проекта

## Хранение данных
- Решение: [выбранная база и способ доступа]
- Почему: [требования, которые она закрывает]
- Компромисс: [известное ограничение]
- Пересмотреть, если: [проверяемое условие]
- Дата: [дата решения]
```

Правило в инструкции агенту:

> Перед изменением архитектуры прочитай DECISIONS.md. Если задача противоречит записанному решению, покажи противоречие и уточни, какое требование изменилось. После согласованного изменения обнови соответствующее решение и его причину.

Храни принятые решения, а не весь диалог. DECISIONS.md - договорённость команды; без указания прочитать файл не стоит рассчитывать, что любой инструмент найдёт его автоматически.

Скачать DECISIONS.md.

МАТЕРИАЛ 25

ИИ пишет «готово», хотя баг остался

Материал к раннему ролику «Сначала тест, потом исправление».

1. Опиши точные действия, ожидаемый результат и то, что происходит сейчас.
2. Попроси воспроизвести ошибку на текущей версии.
3. Создай минимальную проверку, которая падает именно из-за этого бага.
4. Исправь причину и повтори ту же проверку.
5. Проверь ближайший рабочий сценарий, чтобы исправление не сломало его.

Промпт:

> Баг: [ШАГИ]. Ожидаю: [РЕЗУЛЬТАТ]. Получаю: [ФАКТ]. Сначала воспроизведи его и напиши минимальный регрессионный тест. Запусти тест до правки и покажи причину падения. Затем исправь исходную причину, повтори тест и связанные проверки. Не ослабляй ожидания теста ради зелёного результата. Если окружение не позволяет воспроизвести баг, скажи, какого наблюдения не хватает.

Падающий тест из-за отсутствующей зависимости ещё не воспроизводит ошибку приложения. Отдельно проверь, что он доходит до нужного поведения.

МАТЕРИАЛ 26

Агент сам находит баг в браузере

Объединённые материалы роликов про проверку сайта через Codex и подключение браузера к Claude Code: один подход, два инструмента.

Запусти приложение, открой тестовый аккаунт и дай агенту короткий пользовательский путь. В среде агента должен быть доступен браузерный инструмент: текстовый чат сам по себе не видит страницу, Console и Network.

Промпт для подключённого браузера:

> Открой [АДРЕС]. Пройди сценарий: [ДЕЙСТВИЯ]. Найди первый шаг, на котором фактический результат расходится с ожидаемым. Проверь страницу, Console и неуспешные запросы Network. Назови наблюдение, предполагаемую причину и файл, который её подтверждает. Исправь минимальный участок, повтори исходный путь и покажи результат до и после. Не утверждай, что проверил браузер, если инструмент недоступен.

Для Claude Code используй доступную в своей среде браузерную интеграцию и тот же сценарий. Установка браузерного CLI отдельно от агента ещё не означает, что агент им воспользовался. Один из вариантов интеграции - agent-browser.

МАТЕРИАЛ 27

Codex должен видеть результат своих правок

Материал к ролику «Проверка интерфейса на двух размерах».

Открой экран на 390×844 и 1440×900. На каждом размере проверь прокрутку, длинные подписи, кнопки, поля формы и открытый диалог. Скриншот показывает вёрстку, но клики и ввод проверяются отдельно.

Промпт:

> Запусти проект и открой [ЭКРАН] на 390×844 и 1440×900. Пройди сценарий [ДЕЙСТВИЯ]. Сделай скриншоты обычного состояния и открытой модалки. Проверь перекрытия, горизонтальную прокрутку, обрезанный текст и доступность основного действия. Исправь найденные дефекты и повтори те же шаги. В отчёте дай изображения и короткий список подтверждённых изменений.

Для воспроизводимой проверки подходят Playwright и фиксированные тестовые данные. Если экран зависит от загрузки, дождись конкретного элемента интерфейса.

МАТЕРИАЛ 28

Дай Codex визуальный тест, а не только референс

Визуальный тест сравнивает текущий экран с утверждённым эталоном. Установку смотри в гайде «Клики в код» выше.

```
import { test, expect } from '@playwright/test';

test('главная страница', async ({ page }) => {
  await page.setViewportSize({ width: 1440, height: 900 });
  await page.goto('http://localhost:3000');
  await expect(page.locator('main')).toBeVisible();
  await page.evaluate(() => document.fonts.ready);
  await expect(page).toHaveScreenshot('home.png', {
    animations: 'disabled',
    maxDiffPixelRatio: 0.01,
  });
});
```

Адрес и признак готовности замени под проект. Один процент здесь - пример допуска, а не универсальная оценка качества.

Сначала получи эталон на утверждённой версии, просмотри его и сохрани. Дальше запускай обычный тест. Не обновляй baseline автоматически после падения: сначала открой Expected / Actual / Diff. Используй одинаковые браузер, ОС, шрифты и данные. Визуальные сравнения Playwright.

Промпт:

> Исправь расхождение с утверждённым home.png. Не обновляй baseline и не увеличивай допуск. Покажи diff и объясни, какие изменения экрана были намеренными.

Скачать home.spec.ts.

МАТЕРИАЛ 29

100 коммитов, один баг: как найти виноватую правку

Сначала сохрани незавершённую работу. Нужны текущий сломанный коммит и известный рабочий. Вместо GOOD_COMMIT подставь его настоящий хеш.

```
git bisect start
git bisect bad HEAD
git bisect good GOOD_COMMIT
```

Git откроет промежуточную версию. Проверь один и тот же баг и введи одну из команд:

```
git bisect good
```

или:

```
git bisect bad
```

Если версия не запускается по другой причине, используй git bisect skip. Для примерно 100 последовательных кандидатов бинарный поиск обычно требует около семи проверок; пропуски и сложная история меняют оценку.

Автоматический вариант, если npm test проверяет нужный баг на всех версиях:

```
git bisect run npm test
git show refs/bisect/bad
git bisect reset
```

Команда проверки должна возвращать 0 для рабочей версии и ненулевой код для сломанной; 125 означает пропуск. Подготовка зависимостей старых версий может потребовать отдельного скрипта. Документация Git bisect.

МАТЕРИАЛ 30

Одна команда найдёт элемент, который ломает мобильную версию

Открой DevTools → мобильный режим → ширина 390 px. Прокрути страницу к левому краю и выполни в Console код с карточки:

```
[...document.querySelectorAll('*')]
  .filter(el => {
    const r = el.getBoundingClientRect();
    return r.left < 0 || r.right > innerWidth;
  })
  .forEach(el => {
    el.style.outline = '3px solid red';
  });
```

Красная рамка выделит элементы, чьи прямоугольники выходят за экран. Среди них могут быть намеренно спрятанное меню или элементы карусели: список даёт кандидатов, а не готовый диагноз. Псевдоэлементы и содержимое закрытых shadow roots этой командой отдельно не обходятся. Геометрию возвращает getBoundingClientRect.

Найди проблемный блок в Elements. Проверь width/min-width, position, отрицательные отступы и transform. После исправления обнови страницу: диагностические рамки исчезнут.

Промпт:

> На ширине 390 px появляется горизонтальная прокрутка. Подсвеченный элемент: [HTML/СЕЛЕКТОР]. Вот его стили и скриншот: [ДАННЫЕ]. Найди исходную причину переполнения. Не добавляй overflow-x: hidden на всю страницу как замену исправлению. Проверь длинный контент и ширины 390 и 1440 px.

Скачать highlight-overflow.js.

МАТЕРИАЛ 31

Тесты прошли - но им можно верить?

Mutation testing проверяет, замечают ли тесты намеренно внесённую ошибку. Для первого опыта хватит одного условия в отдельной рабочей копии.

1. Убедись, что исходные тесты проходят.
2. Временно измени значимое условие, например isAdmin на !isAdmin.
3. Запусти относящиеся к нему тесты.
4. Восстанови исходный код и проверь diff.
5. Если мутация не замечена, добавь проверку нужного поведения и повтори эксперимент.

Зелёный результат может означать пробел в тестах, неисполняемый участок или мутацию, которая не меняет наблюдаемого поведения. Сначала различи эти случаи.

Промпт:

> В отдельной рабочей копии проверь одну значимую мутацию в [МОДУЛЬ]. Сначала запусти исходные тесты. Измени условие, повтори проверки и зафиксируй, какая из них заметила ошибку. Затем обязательно восстанови исходную реализацию, сохранив только осмысленные улучшения тестов. Не оставляй мутацию в итоговом коде.

Для автоматизации больших наборов мутаций смотри Stryker Mutator.

МАТЕРИАЛ 32

Три Codex, три Git worktree

Разные worktree дают отдельные папки и ветки одного репозитория. Выполни из чистого репозитория с существующим коммитом:

```
git worktree add -b task/ui ../project-ui
git worktree add -b task/api ../project-api
git worktree add -b task/tests ../project-tests
git worktree list
```

Открой каждую папку в отдельной сессии агента. Назначь разные задачи и файлы. Для запущенных приложений используй разные порты; зависимости и локальные env-файлы подготовь в каждой папке.

Промпт одному агенту:

> Твоя задача: [ЗАДАЧА]. Рабочая папка: [ПУТЬ]. Меняй только [ОБЛАСТЬ]. Контракт с другими частями: [КОНТРАКТ]. Если нужны изменения вне области, опиши их отдельно. После работы проверь свой сценарий и сделай отдельный коммит.

Собирай ветки по очереди в интеграционной ветке и проверяй общий сценарий после каждого merge. Worktree не устраняет конфликты и не изолирует общую внешнюю базу. Удалять рабочую папку через git worktree remove можно после сохранения её изменений. Документация worktree.

МАТЕРИАЛ 33

Вайбкодинг 2.0: мини-команда агентов

Рабочие роли из ролика: frontend, backend и проверка. Один человек или ведущий агент согласует задачу и собирает результат.

Роль	Вход	Результат
Frontend	Сценарий, макет, контракт API	Экран и состояния загрузки/ошибки
Backend	Схема данных и контракт	Серверная логика и проверки доступа
Проверка	Критерии готовности	Воспроизведение сценария и найденные дефекты

Задание команде:

> Реализуйте [ОДИН СЦЕНАРИЙ]. Сначала согласуйте формат запроса и ответа. Разделите непересекающиеся области файлов. Frontend и backend работают в отдельных ветках. Проверяющий оценивает итоговый пользовательский путь по критериям [СПИСОК]. Один ответственный объединяет изменения и выполняет общую проверку.

Разделение ролей в одном ответе не запускает несколько независимых процессов. Для параллельной разработки нужны отдельные сессии или поддерживаемая средой работа с подагентами. Для маленькой правки одна сессия часто проще.

МАТЕРИАЛ 34

Как убрать ИИ-слоп из дизайна Codex

Вместо «сделай красиво» дай три референса с разными задачами: типографика, сетка, детали. Объясни, что именно взять с каждого.

Затем согласуй общие значения: цвета, размеры текста, шаг отступов, радиусы и состояния кнопок. Применяй одну систему ко всем экранам.

Промпт:

> Вот три референса. Из первого возьми принцип типографики, из второго - сетку, из третьего - устройство форм. Сначала опиши общую систему и внеси её в tokens.css или существующую тему проекта. Не копируй сайты целиком. Используй [ШРИФТЫ И ЦВЕТА]. Исключи [КОНКРЕТНЫЕ НЕЖЕЛАТЕЛЬНЫЕ ПРИЁМЫ]. Собери один экран и покажи его на 390 и 1440 px. Проверь длинные тексты, фокус клавиатуры и контраст.

Выбирай ограничения по своему продукту: запрет градиентов сам по себе не делает интерфейс хорошим. После первого экрана сравни иерархию, плотность и заметность основного действия.

МАТЕРИАЛ 35

Три скилла для прокачки Codex

В ролике были три разных задачи:

1. frontend-design - помощь с визуальным направлением; anthropics/skills.
2. vercel-react-best-practices - разбор React/Next.js по рекомендациям Vercel; vercel-labs/agent-skills.
3. agent-browser - браузерная автоматизация для агента; vercel-labs/agent-browser.

Команды установки из ролика, запускаемые по отдельности в среде с Node.js:

```
npx skills add anthropics/skills
npx skills add vercel-labs/agent-skills
npx skills add vercel-labs/agent-browser
```

В установщике выбери нужный skill и своего агента. Сверь текущие инструкции репозитория: браузерному инструменту может потребоваться отдельная установка runtime. Прочитай SKILL.md перед применением. Skill задаёт рабочий процесс, но не подтверждает, что проект уже проверен.

Первое задание:

> Примени [SKILL] к одной задаче: [ЗАДАЧА]. Сохрани ограничения проекта. Покажи конкретный результат и проверку, которую удалось выполнить.

МАТЕРИАЛ 36

Три неочевидных скилла для разработки

Подборка из mattpocock/skills:

Skill	Для чего
grill-with-docs	Уточнить замысел и зафиксировать термины и решения
improve-codebase-architecture	Найти участки архитектуры, которые стоит улучшить
handoff	Передать контекст следующей сессии

```
npx skills add mattpocock/skills
```

Выбери нужные навыки. Затем проверь текущий README: в репозитории есть отдельная начальная настройка инженерного процесса, а способ вызова зависит от агента.

Промпт для передачи задачи:

> Подготовь handoff: цель, принятые решения, изменённые файлы, выполненные проверки с результатами, незавершённые пункты и следующий шаг. Отдельно укажи предположения, которые ещё не подтверждены. Не добавляй секреты и не называй незапущенные тесты успешными.

МАТЕРИАЛ 37

ИИ начинает проект с пустой папки

К ролику с обложкой «Тебе больше не нужен GitHub». Уточнение: для локального старта удалённый репозиторий необязателен. GitHub остаётся полезен для совместной работы, удалённой копии и автоматизации; Git и GitHub - разные вещи.

1. Создай пустую папку и открой её в агенте с доступом к файлам.
2. Опиши одного пользователя и один законченный сценарий.
3. Получи минимальный проект и команду запуска.
4. Проверь результат локально и сохрани рабочее состояние в Git.

Промпт:

> Создай в этой пустой папке минимальный проект на [СТЕК]. Пользователь [КТО] должен [ОДНО ДЕЙСТВИЕ]. Нужны только [ЭКРАНЫ]. Подготовь зависимости, запуск и пример данных. Реализуй рабочий сценарий, запусти доступные проверки и объясни, как открыть результат. Не добавляй функции, которых нет в задаче.

МАТЕРИАЛ 38

Убери YouTube Shorts своим расширением

Это локальное расширение Chrome, которое скрывает блоки Shorts в твоём браузере. Оно не удаляет контент с YouTube.

Промпт:

> Создай Chrome Extension Manifest V3. На www.youtube.com скрывай пункт Shorts, полки Shorts на главной и результаты со ссылками /shorts/. Используй content.js и MutationObserver с ограничением частоты обработки. Не скрывай обычные видео и не запрашивай доступ ко всем сайтам. Выдай manifest.json и content.js, объясни локальную установку и отключение. Учти динамическую навигацию YouTube.

Открой chrome://extensions, включи режим разработчика, нажми «Загрузить распакованное расширение» и выбери папку с manifest.json. Обнови вкладку YouTube. Проверь главную, поиск и переходы внутри сайта. Если разметка YouTube изменится, селекторы потребуется обновить. Локальная установка описана в руководстве Chrome Extensions.

МАТЕРИАЛ 39

Пять проектов для первого вечера с вайбкодингом

Ранняя карусель предлагала пять направлений. Время и цена зависят от объёма; для первого вечера ограничься демонстрацией одного сценария.

Идея	Минимальный результат	Что оставить на потом
Лендинг	Предложение, примеры, рабочая форма	Личный кабинет и сложная CMS
Telegram-бот	Одна команда и один полезный ответ	Платежи и десятки состояний
Расширение Chrome	Одно действие на одном сайте	Поддержка множества сайтов
Сервис с AI	Один входной формат и проверяемый результат	Много моделей и сложная очередь
Личный трекер	Добавить запись и увидеть историю	Социальные функции и синхронизация

Промпт:

Хочу сделать [ИДЕЯ]. Сократи её до одного сценария, который можно показать другому человеку. Выпиши входные данные, действие, результат и признак успеха. Предложи минимальную реализацию без необязательных сервисов. В конце дай три шага ручной проверки.

МАТЕРИАЛ 40

Как заработать на вайбкодинге без своего стартапа

Пять услуг из ролика: лендинг, бот, автоматизация рутины, дашборд и небольшое расширение. Услугу проще объяснить через конкретную работу, которую клиент перестанет делать вручную.

Услуга	Что показать в демо
Лендинг	Посетитель понимает предложение и оставляет заявку
Бот	Типовой вопрос получает полезный ответ
Автоматизация	Заявка проходит из формы в нужное место
Дашборд	Данные из таблиц собираются в одну понятную сводку
Расширение	Повторяющееся действие выполняется одной кнопкой

Шаблон предложения клиенту:

> Сейчас вы [РУТИНА]. Могу собрать [ИНСТРУМЕНТ], чтобы [ПРОВЕРЯЕМЫЙ РЕЗУЛЬТАТ]. Первый объём: [СПИСОК]. Демо покажет [СЦЕНАРИЙ]. Для запуска нужны [ДАННЫЕ/ДОСТУПЫ]. Отдельно согласуем поддержку и стоимость внешних сервисов.

Сначала проверь, что задача существует и у неё есть владелец. Зафиксируй результат, сроки и критерий приёмки до разработки. Не обещай доход или экономию, которые ещё не измерены.

МАТЕРИАЛ 41

Собери профиль контекста ME.md

К ролику «Собери цифрового двойника в ChatGPT». ME.md - короткий профиль для ответов, а не точная модель личности.

```
# ME.md

## Кто я
[Роль и опыт, которые важны для моих задач]

## Цели на три месяца
[1-3 конкретные цели]

## Как со мной говорить
[Язык, объём, тон, формат]

## Чего избегать
[Нежелательные приёмы и ограничения]

## Текущие задачи
[Только контекст, нужный для работы]

## Обновлено
[Дата]
```

Приложи файл к задаче или добавь в материалы проекта, если твоя среда это поддерживает.

Инструкция:

> Перед ответом используй ME.md как контекст моих предпочтений. Не соглашайся автоматически; объясняй существенные возражения. Если данных для решения не хватает, задай один конкретный вопрос. После изменения моих целей предложи поправить соответствующий пункт файла.

Храни только то, что помогает работе. Отдельно проверь в новом чате, доступен ли ему файл; само имя ME.md не включает автоматическую память.

Скачать ME.md.

МАТЕРИАЛ 42

Паспорт контекста для нового чата

Если каждый новый диалог начинается с долгих объяснений, собери одну страницу CONTEXT.md.

Сначала интервью:

> Задай мне по одному вопросы, чтобы составить краткий контекст для будущих задач. Выясни пять блоков: кто я, что делаю, как писать, чего избегать, что сейчас важно. Не придумывай ответы. После интервью собери одну страницу и отдельно перечисли неясности.

Структура файла:

```
# CONTEXT.md
## Кто я
## Что делаю
## Как писать
## Чего избегать
## Что сейчас важно
```

В новом чате приложи файл и напиши: «Прочитай CONTEXT.md, затем выполни задачу [ЗАДАЧА]. Если в задаче есть обновлённые данные, используй их и отметь, что стоит изменить в файле».

ME.md из соседнего гайда больше о стабильных предпочтениях. CONTEXT.md удобно использовать как краткий снимок текущей работы. Для одной задачи часто достаточно одного из этих файлов.

Скачать CONTEXT.md.

МАТЕРИАЛ 43

Один промпт, три роли, один цельный ответ

Схема из ролика: архитектор делает черновик, рецензент ищет слабые места, редактор собирает итог. Это последовательные ракурсы одного ответа, а не доказательство независимой экспертизы.

Промпт:

> Задача: [ЗАДАЧА]. Ограничения: [ОГРАНИЧЕНИЯ]. Сначала подготовь решение как архитектор: цель, структура и необходимые действия. Затем оцени его как рецензент: найди пять конкретных слабых мест и недоказанных предположений. Наконец, как редактор собери цельный итог с учётом критики. Покажи краткий список существенных правок и оставшихся вопросов. Не придумывай факты ради полноты.

Лучше работает на ограниченной задаче: структура лендинга, план MVP, письмо или сценарий демо. Проверяемые факты всё равно сверяй с источниками.

МАТЕРИАЛ 44

Идея под тройным допросом

Три ракурса из ролика: практик, скептик и клиент. Каждый оценивает один и тот же план по разным критериям.

Промпт:

> Вот мой план без украшений: [ПЛАН]. Разбери его с трёх сторон. Практик: что сломается при выполнении и какие ресурсы забыты. Скептик: где нет доказательств и какое наблюдение изменит вывод. Клиент: за что он не захочет платить и как решает задачу сейчас. Для каждого замечания укажи способ проверки. Заверши вердиктом: продолжать, сузить или отложить - с условиями, при которых вывод изменится.

Не проси модель угадывать реакцию рынка по одному абзацу. Список возражений - материал для проверки реальными разговорами и наблюдениями.

МАТЕРИАЛ 45

Проверь бизнес-идею до разработки

В ролике предлагался маленький эксперимент с бюджетом до $50. Можно начать и без платной рекламы: цель - проверить конкретную гипотезу спроса.

1. Опиши покупателя, проблему, текущее решение и предполагаемую цену.
2. Найди пять причин, почему продукт могут не купить.
3. Для каждой назови факт, который подтвердит или опровергнет риск.
4. Выбери один тест без разработки и заранее задай критерий результата.

Промпт:

> Идея: [ИДЕЯ]. Покупатель: [КТО]. Найди пять причин, почему это не купят. Для каждой предложи проверку за 48 часов. Затем собери один тест с бюджетом не больше [БЮДЖЕТ]: что показать, где найти 20 релевантных людей, какое действие считать сигналом спроса и при каком результате остановиться. Не путай лайки с готовностью купить. Не придумывай результаты интервью.

Числа 20 и 48 часов - рамка небольшого опыта, не статистическое доказательство рынка. Записывай реальные ответы и действия отдельно от своих выводов.

МАТЕРИАЛ 46

Преврати расписание в календарь с ChatGPT

Приложи фотографию, PDF или таблицу расписания. Сначала получи текстовую расшифровку, затем календарный файл.

Промпт:

> Перенеси это расписание в таблицу: название, дата, начало, конец, место. Часовой пояс: [ГОРОД/ПОЯС]. Период: [ДАТЫ]. Если ячейка размыта, время не указано или неясно повторение, спроси. После моей проверки создай ICS с уникальными UID, временем в указанном часовом поясе и напоминанием за 30 минут. Не добавляй события в мой календарь автоматически.

Проверь таблицу и импортируй ICS сначала в отдельный тестовый календарь. Сверь один обычный день, событие у полуночи и повторения. Повторный импорт того же файла может создать дубликаты в некоторых календарях.

Спецификация формата - iCalendar / RFC 5545. Если в исходнике только дни недели, обязательно укажи дату начала, окончания и исключения.

МАТЕРИАЛ 47

Где зрители уходят: график удержания и таймкоды

Понадобятся график удержания ролика и расшифровка с таймкодами. Скриншот графика без точных значений позволяет увидеть лишь примерные моменты.

Промпт:

> Сопоставь график удержания и расшифровку [МАТЕРИАЛЫ]. Выдели три заметных снижения, укажи примерный таймкод и фразу рядом. Отдели наблюдение от гипотезы о причине. Для каждого участка предложи новую фразу примерно той же длительности, сохрани мой стиль и переход к следующей мысли. Не обещай рост просмотров и не выдумывай значения графика.

Шаблон заметки:

Таймкод	Что видно на графике	Фраза в ролике	Гипотеза	Новая версия
[время]	[наблюдение]	[текст]	[предположение]	[правка]

Меняй ограниченное число участков. На следующем ролике сравни первые секунды и досмотры с учётом длительности и источника трафика. График не объясняет мотив каждого зрителя.

МАТЕРИАЛ 48

Как найти ответ в большом PDF через ИИ

К ролику «ИИ читает PDF за секунды». Скорость и полнота зависят от файла, сканов, таблиц и возможностей выбранного чата.

1. Приложи PDF к инструменту, который поддерживает работу с файлами.
2. Задай один конкретный вопрос по документу.
3. Попроси страницу и короткий подтверждающий фрагмент.
4. Открой указанное место и сверь ответ.

Промпт:

> Ответь на вопрос [ВОПРОС] только по приложенному PDF. Укажи номер страницы файла и, если отличается, напечатанный номер страницы. Дай короткую цитату и объясни вывод. Если ответа нет или страница не читается, прямо скажи об этом. Не подменяй содержание документа знаниями из интернета.

Для сканированного документа может понадобиться распознавание текста. Таблицы, формулы и сноски проверяй по изображению страницы. Если вопрос требует подсчёта, попроси показать исходные строки и способ вычисления.

МАТЕРИАЛ 49

ИИ сам делает презентации: от текста к слайдам

Сначала подготовь структуру, затем проси PPTX или используй редактор презентаций с AI. Возможность экспортировать файл зависит от инструмента; название сервиса на старой карточке не гарантирует его нынешнюю доступность.

Промпт:

> Сделай презентацию на 10 слайдов по материалам [ТЕКСТ/ФАЙЛЫ]. Аудитория: [КТО]. После просмотра она должна [ДЕЙСТВИЕ/ВЫВОД]. На каждом слайде одна мысль, заголовок-вывод и только нужные подтверждения. Не выдумывай цифры и кейсы. Добавь источники рядом с фактами. Сначала предложи план из 10 заголовков. Затем создай редактируемый PPTX, если твоя среда поддерживает экспорт, и проверь переносы, таблицы и читаемость.

При просмотре проверь порядок аргументов, размер текста, источники и отсутствие повторов. PDF удобен для чтения, PPTX - для дальнейшего редактирования.

МАТЕРИАЛ 50

ИИ превращает описание в сайт

Хорошее задание описывает пользователя и действие. Одного пожелания «современный красивый сайт» для этого мало.

Промпт:

> Сделай сайт для [АУДИТОРИЯ]. Главная задача посетителя: [ДЕЙСТВИЕ]. Обязательные блоки: [СПИСОК]. Вот утверждённые тексты и материалы: [ДАННЫЕ]. Стиль: [РЕФЕРЕНСЫ И КОНКРЕТНЫЕ ПРАВИЛА]. Реализуй рабочую навигацию и мобильную версию. Если форма пока не подключена, явно покажи её состояние и не имитируй успешную отправку. Дай адрес предпросмотра и перечень того, что нужно для публикации.

После первого результата проверь каждую кнопку, форму, ссылку и длинный текст. Опубликованный адрес открывай с другого устройства. Превью картинки не заменяет рабочий сайт.

МАТЕРИАЛ 51

Из селфи в редакционный портрет

К раннему ролику о портретах через генератор изображений. Возьми чёткое своё фото без фильтра, с видимым лицом и нормальным освещением.

Промпт из ролика:

> Используй моё фото как референс внешности. Создай реалистичный editorial-портрет: тёмная студия, мягкий боковой свет, впечатление съёмки на объектив 85 mm, натуральная кожа. Сохрани черты лица, возраст и пропорции.

После первой версии меняй одно: свет, фон, одежду или кадрирование. Пример правки: «Сохрани лицо, позу и одежду; сделай только фон светло-серым».

Сравни лицо с исходником, затем проверь руки, волосы, очки и мелкие детали. Даже с референсом генератор может изменить внешность; выбирай результат по сходству, а не только по эффектному свету.

МАТЕРИАЛ 52

Как убрать ИИ-стиль из текста

Дай свой черновик и, если есть, короткий пример того, как ты обычно пишешь.

Промпт:

> Перепиши этот текст естественно. Убери канцелярит, повторы, шаблонные связки и лишний пафос. Сохрани мой смысл, тон, факты, имена и числа. Не добавляй новые обещания. Не дроби текст на однословные строки. Если фраза звучит неестественно, замени её. Текст: [ЧЕРНОВИК]. Пример моего стиля: [ПРИМЕР].

Сравни результат с исходником: не пропали ли важные условия, не изменились ли цифры, не стали ли обещания сильнее. Прочитай вслух и верни характерные слова, которые подходят твоей аудитории.

МАТЕРИАЛ 53

Поиск только по выбранным сайтам

Составь список допустимых доменов и вопрос. Если инструмент умеет ограничивать источники, задай список в его настройках; дополнительно продублируй его в запросе.

Промпт:

> Изучи [ТЕМА]. Используй только эти сайты: [ДОМЕНЫ]. Открывай страницы, которые подтверждают выводы, указывай даты и давай прямые ссылки рядом с фактами. Не используй другие домены для заполнения пробелов. Если на этих сайтах нет ответа, перечисли, что установить не удалось. Отделяй цитату, пересказ и собственный вывод.

Для технического вопроса начни с документации разработчика и исходного репозитория. Проверь домен каждой ссылки в ответе. Одна фраза в промпте не является техническим запретом доступа к остальному интернету и не делает найденное утверждение верным.

МАТЕРИАЛ 54

Пять ошибок вайбкодинга, которые ломают проект

Материал к ранней карусели про секреты, права, базу, оплату и выпуск изменений.

Ошибка	Что проверить
Секретный API-ключ во frontend	Не попадает ли секрет в bundle, исходники страницы и ответы клиенту
Спрятанная кнопка вместо прав	Отказывает ли сервер гостю и пользователю без разрешения
Разрушительная миграция без восстановления	Есть ли свежая копия и проверенный способ вернуть данные
Оплата считается успешной по редиректу	Проверяется ли событие оплаты на сервере по инструкции провайдера
Изменения сразу в рабочем приложении	Есть ли отдельная ветка, preview, проверка и откат

Промпт:

> Проверь проект по этим пяти пунктам. Для каждой проблемы покажи конкретный файл, сценарий воспроизведения и минимальное исправление. Не выводи значения секретов. Не проводи реальные платежи и не запускай разрушительные миграции. После исправления проверь отказ в запрещённом сценарии и успех в разрешённом.

Наличие готовой библиотеки не отменяет правильную настройку. Серверные правила авторизации сверяй с OWASP Authorization Cheat Sheet.

МАТЕРИАЛ 55

Фразы, после которых ИИ перестаёт поддакивать

По мотивам ролика из примера: проси проверяемую критику конкретной мысли.

1. **Отдели факты от догадок.** «Пометь каждое существенное утверждение как факт, предположение или неизвестное. Для факта покажи основание».
2. **Попробуй опровергнуть.** «Приведи сильнейший аргумент против моей идеи и объясни, при каких условиях он верен».
3. **Найди три слабых места.** «Для каждого укажи риск, наблюдение и проверку».
4. **Не соглашайся автоматически.** «Если вывод не следует из данных, покажи, какой шаг рассуждения не подтверждён».
5. **Назови условие пересмотра.** «Какой результат заставит изменить текущий вывод?».

Общий промпт:

> Вот мой вывод: [ВЫВОД]. Вот данные: [ДАННЫЕ]. Отдели факты от предположений, попробуй опровергнуть вывод и найди три слабых места. Не спорь ради спора: если аргумент выдерживает проверку, скажи почему. Для каждого возражения предложи способ проверки. Заверши списком того, чего пока нельзя уверенно утверждать.

Требование спорить не делает ответ точнее само по себе. Проверяй источники и не принимай уверенный тон за доказательство.

МАТЕРИАЛ 56

Пять признаков AI-сайта, которые сразу бросаются в глаза

Материал к показанной карусели «5 признаков AI-сайта». Это признаки шаблонной сборки, а не способ доказать, кто писал код.

Признак	Что исправить
Всё помещено в одинаковые карточки	Выбери структуру под содержание: список, таблицу, текст или отдельный акцент
Все отступы одинаковые	Группируй связанные элементы плотнее, разделы - свободнее
Текст ни о чём	Назови пользователя, конкретную пользу и действие
Мобильная версия сломана	Проверь 390 px, длинные подписи, меню и формы
Нет состояний	Покажи загрузку, пустой результат, ошибку и успех там, где они нужны

Промпт:

Проверь экран [ЭКРАН] по пяти пунктам из таблицы. Для каждого найденного дефекта покажи конкретный элемент и объясни, как он мешает пользователю. Сохрани узнаваемость продукта. Замени только неудачные решения, затем проверь реальный сценарий и мобильную ширину. Не выдавай декоративную переработку за улучшение без объяснения.

МАТЕРИАЛ 57

Этих отзывов никто не писал: проверь AI-сайт

Если ты не давал агенту настоящие отзывы, цифры и клиентские логотипы, проверь, что оказалось на опубликованной странице. Демоданные помогают собрать макет, но не должны выглядеть как подтверждённый опыт клиентов.

Три места для проверки:

1. Отзывы и логотипы. Кто оставил отзыв, где оригинал, можно ли его публиковать? Не выдавай вымышленные имена за покупателей.
2. Числа. «10 000 клиентов», «99% точности», «№1» - какое измерение это подтверждает и за какой период?
3. Обещания. Совпадают ли скорость, результат, условия доступа и гарантии в тексте с тем, что продукт реально делает?

Полный промпт для Codex:

> Проверь публичные тексты этого проекта. Найди отзывы, имена клиентов, клиентские логотипы, рейтинги, количество пользователей, проценты точности, заявления «№1», гарантии и обещания результата. Для каждого пункта покажи текст, путь к файлу, номер строки и найденный источник подтверждения. Если в материалах проекта подтверждения нет, пометь «нужно проверить», а не утверждай, что текст ложный. Отдельно укажи явно вымышленные seed/demo-данные. Предложи конкретную замену: убрать блок, оставить пометку демонстрации в макете или показать реальную функцию продукта. Не придумывай отзывы, цифры, компании и новые обещания. После согласования исправь тексты, проверь вёрстку и перечисли оставшиеся неподтверждённые утверждения.

Таблица проверки:

Текст на сайте	Файл	Источник	Решение
[утверждение]	[путь и строка]	[ссылка или «не найден»]	[оставить / проверить / убрать]

Если подтверждений пока мало, покажи реальный экран продукта, понятный пользовательский сценарий и точные условия использования. Это полезнее для решения посетителя, чем выдуманная статистика.