

Полный отчёт за 13 января 2026

ВКРАТЦЕ

Сегодня был насыщенный день с 45+ коммитами в репозиторий. Основные направления работы:

Улучшения Phase A прототипа:

- Система motion-анимаций для всех интерактивных элементов
- Интеграция Unicorn Studio для WebGL-фонов премиум-уровня
- Масштабный редизайн DS2 Attio с persona-картами и схематичными линиями
- Улучшение всех 5 дизайн-систем до wireframe-accurate состояния
- Исправление критических багов (hydration, TypeScript, React Hooks)
- Добавление Next.js best practices (error handling, SEO, security headers)

Phase B эксперименты:

- Построил два полных прототипа на shadcn/ui (14 секций и 10 секций)
- Результат: чистый shadcn даёт качество 4/10 против 7/10 у Phase A
- Вывод: чистый shadcn не подходит для маркетинговых страниц, но shadcn-based шаблоны ещё стоит проверить

Инфраструктура:

- Проресерчил и установил Claude Skills + MCP серверы
- Активно использую Firecrawl для анализа референсных сайтов
- Обновил документацию репозитория

Новые открытия:

- Aura.build как потенциально более эффективный подход к разработке
- ShadCN Create для кастомизации компонентов до установки
- Drawbridge для визуальной отладки

Цель к пятнице: не просто выбор инструментов, а готовая homepage с финальным дизайном.

Предложение на завтра: начать день с созвона, показать всё что сделано, получить фидбек.

Введение: Напоминание о методологии проекта

Что такое Phase A

Phase A это первый этап нашего проекта, который уже завершён и развёрнут на Vercel по адресу <https://adapty-prototype.vercel.app>

Что это: Прототип homepage Adapty, построенный на чистом Tailwind CSS (без UI-библиотек). В прототипе реализованы 5 разных дизайн-систем, вдохновлённых референсными сайтами: Linear, Attio, Polar.sh, Vercel и Clerk. В интерфейсе есть переключатель в правом верхнем углу, который позволяет моментально переключаться между стилями и сравнивать их.

Зачем это нужно: Цель Phase A была определить, какой визуальный стиль лучше работает для Adapty. Вместо того чтобы угадывать или спорить о предпочтениях, мы построили все варианты и можем сравнить их на реальном контенте.

Результат: Прототип работает, все 14 секций homepage реализованы для каждой из 5 дизайн-систем. Качество оцениваю на 7 из 10. Есть куда расти в плане полировки, но база солидная.

Что такое Phase B

Phase B это второй этап, которым я занимался сегодня.

Что это: Тестирование UI-библиотек поверх выбранного визуального стиля.

Зачем это нужно: Гипотеза была в том, что готовые компоненты из библиотеки дадут более полированный результат за меньшее время. Вместо написания каждого элемента с нуля, берёшь готовый компонент и настраиваешь под свои нужды.

Какую библиотеку выбрал: shadcn/ui. Выбор был логичным по нескольким причинам:

1. Позиционируется как "компоненты, которые ты копируешь к себе", то есть полный контроль над кодом
2. Самая популярная библиотека в React-экосистеме на данный момент
3. Есть активное сообщество и хорошая документация
4. Есть интеграция с Claude через MCP (Model Context Protocol), что ускоряет разработку
5. Ты сам рекомендовал изучить shadcn-экосистему

Результат: Подробнее в разделе про оценку Phase B, но вкратце: чистый shadcn/ui не подошёл для маркетинговых страниц.

Почему мы тестируем инструменты системно

Хочу отдельно объяснить этот момент, потому что может показаться, что "мы всё ещё ищем инструменты" это что-то плохое. На самом деле наоборот.

Альтернативный сценарий: Представь, что мы бы просто взяли shadcn/ui на основе его популярности и начали строить весь сайт. Через неделю работы стало бы понятно, что результат получается generic и не дотягивает до уровня Linear или Attio. Пришлось бы переделывать. Потерянное время, потерянная мотивация, сдвинутые дедлайны.

Что мы делаем вместо этого: Я трачу один день на системный тест. Строю полноценные прототипы, сравниваю результаты количественно (7/10 vs 4/10), и получаю обоснованное решение вместо догадки.

Почему это экономит время: Сегодняшний день тестирования shadcn это не "трата времени на поиски". Это сэкономленная неделя переделок. Теперь я точно знаю, что чистый shadcn не подходит, и могу двигаться дальше с уверенностью.

Раздел 1: Оценка Phase B (shadcn/ui)

ВКРАТЦЕ

Чистый shadcn/ui не подходит для маркетинговых страниц. Эта библиотека создавалась для дашбордов и приложений, где важна консистентность UI-элементов. Для landing pages, где нужен контроль над каждым пикселем, базовые компоненты добавляют лишнюю работу без прироста качества.

Результат: Phase A на чистом Tailwind дал качество 7/10. Phase B на чистом shadcn дотянул только до 4/10.

Но это не полный отказ от shadcn-экосистемы. Есть shadcn-based шаблоны (например, Tailark marketing blocks), которые специально созданы для landing pages. Их ещё нужно проверить.

Контекст эксперимента

Что я сделал: Построил два полных прототипа на shadcn/ui с нуля:

- Proto-DS1-Linear: тёмная тема, premium стиль, 14 из 14 секций
- Proto-DS2-Attio: светлая тема, editorial стиль, 10 из 14 секций

Как строил: Использовал shadcn/ui MCP для Claude, который позволяет быстро генерировать компоненты. Комбинировал базовые shadcn компоненты (Button, Card, Badge, etc.) с кастомными стилями.

Сколько времени заняло: Около 6 часов суммарно на оба прототипа.

Что показал эксперимент

Главная проблема: shadcn создавался для дашбордов и SaaS-приложений. Там нужна консистентность UI-элементов: кнопки должны выглядеть одинаково везде, формы работать предсказуемо, модальные окна открываться по стандартному паттерну. Для маркетинговых страниц требования совсем другие.

Почему это проблема для landing pages: На landing page каждая секция уникальна:

- Hero-секция должна впечатлять и выделяться
- Feature-блоки должны рассказывать историю продукта
- Testimonials должны вызывать доверие
- Pricing должен быть понятным и привлекательным

Здесь нет "стандартных компонентов". Каждый элемент кастомизируется под конкретную задачу. Кнопка в Hero может выглядеть совсем иначе, чем кнопка в footer.

Что происходит на практике: Когда я начал строить прототипы на shadcn, столкнулся с постоянной необходимостью переопределять стили. Базовые компоненты выглядят слишком generic. Чтобы добиться нужного визуала, приходится писать override за override:

- Менять padding и margin
- Переопределять цвета
- Менять типографику
- Добавлять кастомные hover-эффекты
- Переписывать структуру компонента

В итоге работы столько же, сколько при создании с нуля, но структура кода получается сложнее из-за дополнительного слоя абстракции.

Конкретные результаты

Proto-DS1-Linear на shadcn:

- Секций реализовано: 14 из 14 (100%)
- Качество визуала: 4 из 10
- Проблема: компоненты выглядят как "ещё один сайт на shadcn", теряется уникальность Linear-inspired дизайн-системы. Тёмная тема получилась плоской, без глубины и 3D-эффектов, которые есть на оригинальном Linear.

Proto-DS2-Attio на shadcn:

- Секций реализовано: 10 из 14 (71%)
- Качество визуала: 4 из 10
- Проблема: даже с существенной кастомизацией результат не дотягивает до референса. Светлая тема выглядит generic, без editorial-feel оригинального Attio.

Phase A для сравнения:

- Секций реализовано: 14 из 14 для каждой из 5 дизайн-систем
- Качество визуала: 7 из 10
- Преимущество: полный контроль над каждым элементом, возможность экспериментировать без ограничений библиотеки

Вывод и следующие шаги

Вывод: Чистый shadcn/ui не подходит для маркетинговых страниц Adapty.

Где shadcn всё ещё имеет смысл:

- Дашборды и админки
- Внутренние инструменты
- Формы и функциональные UI-элементы (например, форма Schedule a Demo)

Что ещё стоит проверить: Мы не полностью отказываемся от shadcn-экосистемы. Есть shadcn-based шаблоны и component libraries, которые специально созданы для marketing pages:

- Tailark marketing blocks (<https://tailark.com/docs>)
- Другие shadcn-based landing page kits

Эти инструменты могут работать лучше, потому что они изначально оптимизированы под landing pages, а не под дашборды. Их нужно проверить.

Раздел 2: Улучшения Phase A прототипа

ВКРАТЦЕ

Параллельно с оценкой shadcn продолжал развивать Phase A прототип. За день сделал значительные улучшения:

- Система motion-анимаций для всех интерактивных элементов
- Интеграция Unicorn Studio для WebGL-фонов
- Масштабный редизайн DS2 Attio
- Улучшение всех 5 дизайн-систем до wireframe-accurate состояния
- Исправление критических багов
- Добавление Next.js best practices

Эта работа подтвердила главное: vanilla Tailwind даёт полную свободу для экспериментов и позволяет достичь качества, которого сложно добиться с библиотеками.

Motion-анимации

Что это: Полноценная система анимаций для всех интерактивных элементов в прототипе.

Что конкретно добавил:

- Hover-эффекты на кнопках (scale, background transition)
- Появление секций при скролле (fade-in, slide-up)
- Subtle feedback на карточках при взаимодействии
- Плавные переходы при переключении дизайн-систем

Как реализовал: Комбинация CSS-анимаций для простых эффектов и Motion library (бывший Framer Motion) для более сложных. Motion library даёт больше контроля над timing и easing.

Зачем это нужно: Анимации делают интерфейс более живым и premium. Это один из ключевых элементов, который отличает сайты уровня Linear от обычных landing pages. Пользователь чувствует, что каждый элемент реагирует на его действия.

Результат: Прототип стал значительно более интерактивным. Переходы плавные, кнопки отзывчивые. Особенно заметно на DS1 (Linear-inspired), где motion-design играет ключевую роль.

Unicorn Studio интеграция

Что это: Unicorn Studio это сервис для создания WebGL-фонов без написания shader-кода. Работает через embed'ы, которые вставляются в React-компоненты.

Что конкретно сделал:

- Встроил первый embed как фон для Hero-секции в DS1 (Linear-inspired). Это градиентный mesh с частицами, реактивный на движение курсора.
- Встроил второй embed как отдельный декоративный блок перед footer в DS3 (Polar-inspired).
- Исправил несколько технических проблем интеграции: props validation (убрал невалидный style prop), lazy loading (отключил для немедленного рендера), размеры (добавил explicit width/height).

Зачем это нужно: WebGL-эффекты дают premium-ощущение, которого сложно добиться чистым CSS. На оригинальном сайте Linear активно используются подобные эффекты. Они создают глубину и атмосферу.

Почему Unicorn Studio: Писать shader-код с нуля это недели работы специалиста. Unicorn Studio даёт доступ к готовым эффектам уровня Linear за минуты. Сервис платный, но стоимость оправдана экономией времени.

Результат: Hero-секция в DS1 теперь имеет animated gradient background, который реагирует на курсор. Выглядит значительно более premium. В DS3 добавлен декоративный блок, который разбавляет минималистичный дизайн.

Масштабный редизайн DS2 Attio

Что это: Полная переработка дизайн-системы Attio в прототипе, чтобы она точнее соответствовала оригинальному сайту.

Что конкретно добавил:

1. Persona-карты вместо стандартных feature-карточек:

- Оригинальный Attio использует карточки с "персонами" (иконки людей, роли, описания)
- Заменял generic feature-карточки на persona-style cards
- Результат: более editorial feel, как на оригинале

2. Схематичные соединительные линии:

- На сайте Attio есть тонкие линии, соединяющие элементы
- Добавил connector line system с SVG-линиями
- Результат: визуальная связность между секциями

3. Точечная сетка (dot grid) как фоновый паттерн:

- Attio использует subtle dot pattern на фоне
- Реализовал через CSS radial-gradient
- Пришлось несколько раз исправлять opacity (с 6% до 15%), чтобы паттерн был заметен, но не отвлекал

4. Аутентичный viewport frame:

- На Attio есть рамка вокруг основного контента
- Добавил border и padding, как на оригинале
- Результат: более "обрамлённый" feel

5. Специальный tab bar:

- На Attio tab bar выглядит иначе, чем стандартные tabs
- Переделал на full-width bordered container
- Результат: точное соответствие оригиналу

6. Специальные варианты Header и FinalCTA:

- Header и footer на Attio имеют свои особенности
- Добавил DS2-specific варианты этих компонентов
- Результат: консистентность дизайн-системы от header до footer

Зачем это нужно: Цель Phase A была протестировать разные дизайн-системы. Но если DS2 не похожа на Attio, тест бессмысленный. Редизайн приблизил прототип к оригиналу, что даёт более честное сравнение.

Результат: DS2 теперь выглядит значительно ближе к оригинальному Attio. Fidelity повысилась с примерно 35% до 85%.

Улучшение всех 5 дизайн-систем

Что это: Дотошный аудит и исправление каждой дизайн-системы, чтобы она точнее соответствовала референсному сайту.

Что конкретно сделал:

1. Исправил цвета primary-кнопок для DS1, DS2, DS4:

- Сравнил с референсными сайтами
- Обнаружил расхождения в оттенках
- Исправил hex-коды на точные значения из оригиналов

2. Реализовал "Design DNA" изменения:

- Внедрил polymorphic components, которые меняют поведение в зависимости от активной DS
- Например, кнопка в DS1 имеет другой hover-эффект, чем в DS5

3. Выровнял ordering секций с аутентичными wireframes:

- Порядок секций на разных референсных сайтах отличается
- Настроил ds-configs.ts так, чтобы каждая DS имела свой порядок секций

4. Улучшил Linear, Attio, Polar, Clerk до wireframe-accurate состояния:

- Провёл side-by-side сравнение с оригиналами
- Исправил spacing, typography, colors
- Добавил недостающие визуальные элементы

5. Исправил проблему с фиксированными background colors в DynamicPage:

- Была ошибка, когда фоны секций не менялись при переключении DS
- Убрал hardcoded значения
- Теперь фоны корректно переключаются

6. Переименовал DS5 из "Hybrid" в "Clerk":

- DS5 изначально называлась "Hybrid Premium"
- Но по факту она inspired by Clerk
- Переименовал для ясности

Зачем это нужно: Качество теста дизайн-систем напрямую зависит от того, насколько точно прототипы воспроизводят оригиналы. Если DS1 не похожа на Linear, как мы можем оценить, подходит ли Linear-style для Adapy?

Результат: Все 5 дизайн-систем теперь значительно ближе к своим референсам. Это делает сравнение более объективным и полезным.

Next.js best practices

Что это: Добавление production-ready функциональности в прототип.

Что конкретно добавил:

1. Error handling:

- Добавил error boundaries

- Graceful fallbacks для компонентов, которые могут сломаться
- User-friendly error messages

2. SEO оптимизация:

- Proper meta tags
- Open Graph images
- Structured data где уместно

3. Security headers:

- Content Security Policy
- X-Frame-Options
- Другие security-related headers

Зачем это нужно: Прототип может в какой-то момент стать production-сайтом. Лучше заложить правильную архитектуру сразу, чем переделывать потом.

Результат: Прототип готов к потенциальному production deployment с точки зрения базовой инфраструктуры.

Раздел 3: Инфраструктура и инструменты

ВКРАТЦЕ

Потратил значительную часть дня на настройку инфраструктуры для ускорения разработки. Проресерчил и установил Claude Skills + MCP серверы. Активно использую Firescrawl для анализа референсных сайтов. Эти инструменты уже окупались в виде ускорения работы.

Claude Skills и MCP серверы

Что это: Claude Skills это специализированные наборы инструкций, которые улучшают работу Claude в конкретных областях. MCP (Model Context Protocol) серверы это интеграции, которые дают Claude доступ к внешним инструментам.

Что конкретно установил и настроил:

1. frontend-design skill:

- Описание: создание distinctive, production-grade frontend interfaces
- Зачем: помогает генерировать более качественный UI-код, который не выглядит как "generic AI slop"
- Как использую: активируется автоматически при работе над UI-компонентами

2. frontend-ui-ux skill:

- Описание: designer-turned-developer mindset
- Зачем: фокус на pixel-perfect детали, spacing, micro-interactions
- Как использую: при полировке визуала и работе с анимациями

3. styling-with-shadcn skill:

- Описание: работа с shadcn/ui компонентами
- Зачем: было нужно для Phase B экспериментов
- Как использую: при работе с shadcn-based прототипами

4. shadcn MCP server:

- Описание: интеграция с shadcn/ui библиотекой
- Зачем: быстрая генерация и кастомизация компонентов

- Как использую: при создании Phase B прототипов

5. 21st.dev Magic MCP:

- Описание: доступ к 21st.dev component library
- Зачем: дополнительные UI-компоненты и inspirations
- Как использую: при поиске интересных UI-паттернов

6. Firecrawl MCP:

- Описание: web scraping и document fetching
- Зачем: анализ референсных сайтов
- Как использую: подробнее в следующем разделе

Результат использования: На основе этих skills и серверов провёл дотошный аудит соответствия прототипов собранным дизайн-системам. С включенными релевантными skills Claude генерирует более качественный код, который требует меньше правок.

Firecrawl

Что это: Firecrawl это сервис для web scraping и document fetching, который мы приобрели вчера.

Как использую:

1. Извлечение CSS-переменных из референсных сайтов:

- Скрейплю CSS из Linear, Attio, Polar, Vercel, Clerk
- Получаю точные значения цветов, spacing, typography
- Использую эти значения для настройки дизайн-систем

2. Анализ структуры контента:

- Извлекаю HTML-структуру страниц
- Понимаю, как организованы секции
- Использую как reference при построении прототипов

3. Извлечение текстового контента:

- Получаю copy и messaging с референсных сайтов
- Анализирую tone of voice и структуру аргументации
- Полезно для понимания, как конкуренты подают информацию

Результат: Firecrawl сильно ускоряет research phase. Вместо ручного копирования CSS-значений из DevTools, получаю structured data за секунды.

Документация

Что это: Обновление документации репозитория для лучшей организации проекта.

Что конкретно сделал:

1. README.md:

- Добавил визуалы и скриншоты
- Добавил badges (build status, deployment status)
- Написал comprehensive описание проекта
- Добавил quick start инструкции

2. CHANGELOG:

- Создал comprehensive changelog всех коммитов
- Организовал по датам и категориям

- Полезно для отслеживания прогресса

3. CLAUDE.md файлы:

- Обновил во всех ключевых директориях
- Добавил контекст для будущих сессий
- Описал текущее состояние и следующие шаги

4. docs/PROGRESS.md:

- Создал полную историю проекта
- Описал все phases и их результаты
- Useful для onboarding и reference

Зачем это нужно: Хорошая документация экономит время в будущем. Когда я открою проект через неделю, не нужно будет вспоминать, что где находится и что было сделано.

Результат: Репозиторий теперь хорошо документирован и организован.

Раздел 4: Новые открытия

ВКРАТЦЕ

Нашёл несколько инструментов, которые могут ускорить разработку:

- Aura.build: AI-генератор шаблонов landing pages
- ShadCN Create: кастомизация shadcn компонентов до установки
- Drawbridge: визуальная отладка с Claude Code

Aura.build выглядит особенно перспективно и будет следующей гипотезой для проверки.

Aura.build

Что это: Aura.build это сервис от @MengTo (Meng To), известного дизайнера и создателя Design+Code. Он использует AI для генерации готовых шаблонов landing pages.

Как это работает:

1. Описываешь, какой сайт нужен
2. AI генерирует готовый шаблон высокого качества
3. Экспортируешь в чистый HTML + Tailwind CSS
4. Конвертируешь в React-компоненты
5. Код полностью твой, нет привязки к сервису

Почему это интересно:

1. Качество шаблонов:

- Позиционируется как шаблоны уровня Linear, Vercel, Attio
- То есть именно того уровня, к которому мы стремимся

2. Экспорт в нативный код:

- В отличие от Webflow или Framer, код полностью твой после экспорта
- Можно редактировать как угодно
- Нет vendor lock-in

3. Экономия времени:

- Вместо построения каждой секции с нуля, берёшь готовый шаблон
- Адаптируешь под бренд Adapty

- В теории это значительно быстрее

Потенциальная комбинация:

- Aura.build для базовой структуры страниц
- Unicorn Studio для WebGL-эффектов
- Motion library для анимаций
- Результат: premium-уровень за fraction of time

Риски и неизвестные:

1. **Качество кода:** Не тестировал реальный output. Может быть много мусорного кода, который придётся вычищать.
2. **Время на адаптацию:** Не знаю, сколько времени уходит на адаптацию шаблона под конкретные нужды. Если это занимает столько же, сколько создание с нуля, смысл теряется.
3. **Гибкость:** Не уверен, насколько хорошо сервис понимает нетривиальные запросы. Простой лендинг он наверняка сделает, но сложную структуру типа Adapty homepage с 14 секциями не факт.

Следующий шаг: Завтра протестирую Aura.build на практике. Попробую сгенерировать шаблон, похожий на структуру Adapty homepage, и оценю:

- Качество кода на выходе
- Время на адаптацию
- Гибкость для наших нужд

ShadCN Create

Что это: Относительно новый инструмент на ui.shadcn.com/themes, который позволяет кастомизировать компоненты shadcn ДО их установки в проект.

Как это работает:

1. Заходишь на ui.shadcn.com/themes
2. Настраиваешь цветовую палитру, шрифты, радиусы скруглений
3. Предпросматриваешь компоненты с твоими настройками
4. Генерируешь install command с твоими preset'ами
5. Все установленные компоненты уже в твоём стиле

Почему это интересно: Решает часть проблемы с generic-видом shadcn. Компоненты сразу выглядят более кастомно, не нужно переопределять стили после установки.

Ограничения: Не решает главную проблему, что shadcn ограничивает свободу кастомизации на уровне отдельных элементов. Для дашбордов и форм этого достаточно, для landing pages нет.

Где может пригодиться: Функциональные части сайта, такие как:

- Форма Schedule a Demo
- Login/signup flows
- Pricing calculator
- Интерактивные элементы

Drawbridge

Что это: Инструмент для визуальной отладки, интегрированный с Claude Code.

Как это работает:

1. Указываешь на элемент интерфейса
2. Говоришь, что нужно изменить

3. Claude понимает визуальный контекст и делает fix

Почему это интересно: Устраняет необходимость описывать проблемы словами. Вместо "подвинь кнопку на 2 пикселя влево и уменьши отступ снизу" можно просто показать на элемент и сказать "это должно быть вот здесь".

Где пригодится: На этапе полировки, когда нужно довести визуал до pixel-perfect состояния. Особенно полезно для:

- Выравнивание элементов
 - Точная настройка spacing
 - Визуальная отладка responsive behavior
-

Раздел 5: Почему ещё нет плана на неделю

ВКРАТЦЕ

Ты просил план на неделю, и я мог бы его написать. Но честно: сейчас это было бы бессмысленно. Мы находимся в точке, где выбор подхода к разработке определяет всё остальное. Сначала нужно определиться с toolset, потом план напишется за час и будет иметь реальный смысл.

Честное объяснение

Ты вчера написал, что ждёшь план. И я понимаю, почему это важно для планирования твоей работы и понимания прогресса.

Но давай честно: сейчас писать план было бы бессмысленно.

Проблема: Мы находимся в точке, где выбор подхода к разработке определяет всё остальное.

Пример: Представим, я напишу план "На этой неделе делаем homepage и pricing page на shadcn/ui". А через два дня выяснится, что shadcn не подходит (что и произошло). Весь план летит в мусорку, нужно переписывать.

Другой пример: Представим план "Делаем homepage на vanilla Tailwind". А завтра окажется, что Aura.build работает и ускоряет разработку в 3 раза. Опять план неактуален.

Почему я сфокусировался на тестировании

Вместо написания плана, который устареет через день, я сфокусировался на том, чтобы получить данные для принятия решения:

1. **Протестировал shadcn:** теперь знаю, что чистый shadcn не подходит
2. **Улучшил Phase A:** теперь есть качественный baseline для сравнения
3. **Нашёл альтернативы:** Aura.build, Tailark, ShadCN Create

Когда мы определимся с toolset, план напишется за час и будет иметь реальный смысл.

Что будет после определения toolset

Как только мы решим, какой подход используем, я смогу написать конкретный план:

1. Если Aura.build работает:

- День 1-2: генерация базовых шаблонов для priority pages
- День 3-4: адаптация под бренд Adapty
- День 5: homepage готов, начинаем pricing

2. Если возвращаемся к Phase A:

- День 1-2: полировка homepage до 9/10

- День 3-4: применение тех же паттернов к pricing
- День 5: две страницы готовы

3. Если Tailark работает:

- Аналогичная структура с shadcn-based компонентами

Но сейчас писать эти планы детально бессмысленно, потому что мы не знаем, какой путь выберем.

Раздел 6: Предложение на завтра

ВКРАТЦЕ

Предлагаю начать завтра с созвона. Покажу всё, что сделано. Объясню логику принятия решений. Послушаю твой фидбек. После этого протестирую оставшиеся гипотезы и начну работу над homerage.

Созвон в начале дня

Хочу начать завтра с созвона с тобой. Вот что я покажу и объясню:

1. Phase A прототип:

- Переключение между 5 дизайн-системами в реальном времени
- Демонстрация motion-анимаций
- Показ WebGL-фонов от Unicorn Studio
- Объяснение, как это работает технически

2. Phase B результаты:

- Покажу прототипы на shadcn
- Объясню, почему качество ниже
- Покажу конкретные примеры проблем

3. Новые инструменты:

- Демо Aura.build (если успею протестировать до созвона)
- Показ ShadCN Create
- Объяснение, как Firecrawl помогает с research

4. Логика принятия решений:

- Почему тестирование это не "трата времени"
- Как это экономит время в долгосрочной перспективе
- Какие гипотезы ещё нужно проверить

5. Вопросы к тебе:

- Какие дизайн-системы нравятся больше?
- Есть ли предпочтения по инструментам?
- Какие приоритеты по страницам?

После созвона

После созвона у меня будет понимание твоих приоритетов и предпочтений. И я смогу:

1. Протестировать Aura.build и/или Tailark:

- Займёт пару часов
- Получу данные для финального решения

2. Определиться с финальным подходом:

- На основе всех тестов и твоего фидбека
- Выбор toolset, который используем дальше

3. Начать работу над homerage:

- С выбранным toolset
 - Цель: готовая homerage к пятнице
-

Раздел 7: Цель к пятнице

ВКРАТЦЕ

К пятнице нужна готовая homerage с финальным дизайном. Это будет centerpiece проекта. Также планирование следующей недели и движение к дедлайну 12 февраля.

Что должно быть готово

К пятнице нужна не просто финализация набора инструментов, а **готовая homerage**.

Что значит "готовая":

- Все 14 секций реализованы
- Правильные шрифты (скорее всего Gilroy или близкий к нему)
- Правильные цвета и стили
- Работающие анимации
- WebGL-эффекты где уместно
- Responsive design
- Визуальное качество 9/10

Зачем это нужно

Эта homerage будет нашим **centerpiece**:

1. **Утверждение подхода:** На ней мы увидим, как выглядит финальный результат с выбранным toolset.
2. **Основа для остальных страниц:** Компоненты и паттерны из homerage будут переиспользоваться на pricing, feature pages и т.д.
3. **Демонстрация progress:** Конкретный результат, который можно показать и обсудить.
4. **Валидация решений:** Если homerage выглядит на 9/10, значит мы на правильном пути.

Планирование в пятницу

Помимо демонстрации homerage, в пятницу спланируем следующую неделю:

1. Приоритетные страницы:

- Homerage (будет готова)
- Pricing
- Schedule a Demo
- Feature pages (paywall-builder, paywall-ab-testing, onboarding-builder)
- Role pages (for-marketers, for-developers, for-app-owners)

2. Распределение работы:

- Сколько страниц в неделю реалистично
- Какие можно делать параллельно

- Где нужны дополнительные ресурсы

3. Движение к дедлайну 12 февраля:

- У нас примерно месяц
 - Нужно перевезти весь сайт
 - Составим roadmap по неделям
-

Приложения

Текущее состояние прототипов

Phase A прототип:

- URL: <https://adapty-prototype.vercel.app>
- Статус: развёрнут на Vercel, работает
- Секций: 14 из 14 для каждой из 5 дизайн-систем
- Качество: 7 из 10 (хорошая база, есть куда полировать)
- Особенности: motion-анимации, WebGL-фоны, переключатель тем

Phase B Proto-DS1-Linear:

- Статус: построен локально
- Секций: 14 из 14
- Библиотека: shadcn/ui
- Качество: 4 из 10

Phase B Proto-DS2-Attio:

- Статус: построен локально
- Секций: 10 из 14
- Библиотека: shadcn/ui
- Качество: 4 из 10

Phase B Proto-DS3-Polar:

- Статус: только scaffolding
- Не развивался, т.к. результаты DS1 и DS2 уже показали проблему

Phase B Proto-DS4-Vercel и Proto-DS5-Clerk:

- Статус: не начаты
- По той же причине

Статистика за день

- Коммитов: 45+
- Часов работы: ~12
- Прототипов построено: 2 (Phase B)
- Прототипов улучшено: 1 (Phase A)
- Инструментов установлено: 6+ (skills + MCP servers)
- Багов исправлено: 6 критических

Ссылки

- Phase A прототип: <https://adapty-prototype.vercel.app>
- Репозиторий: <https://github.com/kirniy/adapty-webdev>
- Aura.build: <https://www.aura.build>
- ShadCN Create: <https://ui.shadcn.com/themes>

- Unicorn Studio: <https://unicorn.studio>
- Drawbridge: <https://github.com/breschio/drawbridge>
- Firecrawl: <https://firecrawl.dev>
- Tailark: <https://tailark.com/docs>

Кирилл, 14 января 2026