Alt-Vows — Complete Technical & Product Audit

> ***Date**: Feb 13, 2026*
> ***Stack**: Next.js 16.1.6 · React 19.2.3 · Supabase · TypeScript · Tailwind CSS v4 · Zustand 5 · Framer Motion 12*

---

1. THEME ENGINE

Themes in the Registry

Theme	`archetypeId`	Status	Sub-pages
Vintage Vinyl	`vintage-vinyl`	✅ Fully built, registered in `registry` + dynamic routing	Gatekeeper → Player (Invite mode), Split-screen LandingPage
The Voyager	`the-voyager`	⚠️ Built but **disabled** in dynamic routing — commented out in [registry.ts]()	LandingGate → FlightTransition → JourneyScroll

Vintage Vinyl — Style Variants (from `niches.json`)

Niche Slug	Vibe	Primary Color	Fonts	Status
`rock-n-roll-wedding`	Dark/Heavy	`#e02e2e` (Red)	Metal Mania / Roboto	✅ Full
`jazz-lounge-wedding`	Warm/Gold	`#d4af37` (Gold)	Playfair Display / Lato	✅ Full
`cyberpunk-wedding`	Neon/Dark	`#00f0ff` (Cyan)	Orbitron	⚠️ Partial — uses an older flat JSON format, missing `hero`/`theme` nesting

How JSON Schema Drives the Renderer

```
graph LR
    A[niches.json] -->|mergeConfig| B[VinylThemeConfig]
    B --> C{layoutMode?}
    C -->|split-screen| D[LandingPage]
    C -->|default| E[Gatekeeper → Player]
    F[Zustand Store] -->|activeTheme| E
    F -->|activeTheme| D
```

1. `niches.json` holds per-niche style variants (colors, fonts, textures, SVG corner shapes)
2. [mergeConfig()]() deep-merges niche data with `defaultContent` to produce a fully typed `VinylThemeConfig`
3. The config drives 3 typed sub-pages: `LandingPage` , `Gatekeeper` , `Player`
4. The Voyager uses a separate `ThemeConfig` interface with a `schema[]` + `defaultContent` structure

Animations / Interactions Built

Feature	Technology	Theme
Gatekeeper → Player page transition	`AnimatePresence` (Framer Motion)	Vintage Vinyl
Vinyl record spinning	CSS animation	Vintage Vinyl
Neon glow / button hover effects	CSS + config-driven colors	Vintage Vinyl
LandingGate → FlightTransition → JourneyScroll	`AnimatePresence` with 3-stage flow	The Voyager
Confetti celebration on payment	CSS `@keyframes`	Global
RSVP Modal spring animation	Framer Motion `spring` transition	Global

---

2. LIVE EDITOR

Architecture

```
graph TD
    A[useEditorStore] -->|Zustand| B[activeTheme state]
    C[EditorSidebar] -->|reads schema| D[vintageVinylSchema]
    C -->|getNestedValue| B
    C -->|updateField path, val| A
    B -->|reactive rendering| E[Theme Component]
```

- **Store**: useEditorStore.ts — Zustand store with `activeTheme` , `isOpen` , `setTheme()` , `toggleEditor()` , `updateField(path, value)`
- `updateField` uses `setNestedValue()` — a deep-clone + dot-path setter (e.g. `hero.title`)
- The editor uses `JSON.parse(JSON.stringify(obj))` for deep cloning — works but is slow for large state objects

Editable Fields

Field Type	Implementation	Used For
`text`	Standard `<input>`	Names, dates, titles, urls
`textarea`	`<textarea>`	Story descriptions
`color`	`<input type="color">` + hex preview	Primary color, glow color
`image`	ImageField — URL paste + file upload	Story image, photos
`image-list`	ImageListField — multi-image grid	Gallery images array
`select`	`<select>` with options	Frame style (minimal/tape/gold)

ImageField vs ImageListField

- **ImageField**: Single image — shows preview, accepts URL paste or file upload, produces a single string value
- **ImageListField**: Array of images — shows a grid of uploaders, produces a `string[]` , allows add/remove

Known Editor Issues

#	Bug	Symptom	Suspected Cause
1	**Editor hardcoded to vintage-vinyl**	The EditorSidebar always imports `vintageVinylSchema` — won't work for The Voyager or future themes	No dynamic schema resolution by theme ID
2	**Save hardcodes `theme_id`**	handleSave always writes `theme_id: 'vintage-vinyl'`	Should read from activeTheme or route context
3	**No debounce on updates**	Every keystroke triggers a full deep-clone via `JSON.parse(JSON.stringify())`	Could cause jank on slower devices

3. AUTHENTICATION & USER FLOW

Implementation

Piece	File	Method
Auth Page	app/auth/page.tsx	Email + password (sign in / sign up tabs)
Auth Context	hooks/useAuth.tsx	React Context with `onAuthStateChange` listener
Client SDK	lib/supabase.ts	Anon key client
Admin SDK	lib/supabase-server.ts	Service role key (bypasses RLS, used in webhooks)

Session Flow

1. User signs up → receives email confirmation → signs in
2. `AuthProvider` wraps entire app via root layout
3. On sign-in, Supabase sets session cookie → `onAuthStateChange` fires → `user` state updated
4. Dashboard and create pages redirect to `/auth` if `user` is null

RLS Policies

> *[!WARNING] **No RLS policies were found in the codebase.** RLS must be configured in the Supabase dashboard. The `websites` table uses `owner_id` for ownership, and the checkout API manually verifies `.eq('owner_id', user.id)`. The webhook uses `supabaseAdmin` (service role) to bypass RLS.*

Assumption: RLS is either disabled or configured in Supabase dashboard. This is a **critical security item** to verify before launch.

4. DASHBOARD

What it Shows

app/dashboard/page.tsx — Protected page showing all user's wedding sites as cards.

Element	Details
Header	"My Wedding Sites" + user email + "+ New Site" + "Sign Out"

Site Cards	Color banner (derived from `theme.global.palette.primary`), couple names, theme name, last edited date
Status Badge	✅ **Live** (green, `status === 'production'`) or ⚡ **Demo** (yellow)
Card Actions	**Edit** (→ `/demo/{slug}`), **Publish $49** (demo sites), **Share Link** (live sites)
Empty State	💿 emoji + "Create Your First Site" CTA
Payment Success	Detects `?payment=success&slug=X` → shows `ConfettiSuccess` overlay

Create Site Flow

app/dashboard/create/page.tsx:

1. Enter couple names → auto-generates slug
2. Pick a wedding date
3. Choose vibe: Rock & Roll 🤘, Jazz Lounge 🎷, Cyberpunk ⚡
4. Creates site in `websites` table with `mergeConfig(nicheConfig)` → redirects to `/demo/{slug}`

---

5. DATABASE SCHEMA

Tables (inferred from code — no migration files found)

`websites` table

Column	Type	Source
`slug`	`text` (PK)	dashboard/page.tsx
`theme_id`	`text`	Archetype ID (e.g. `vintage-vinyl`)
`content`	`jsonb`	Full theme config object
`status`	`text`	`'demo'` (default) or `'production'`
`owner_id`	`uuid` (FK → auth.users)	Supabase auth user ID
`payment_id`	`text`	LemonSqueezy order ID
`created_at`	`timestamptz`	Auto-generated
`updated_at`	`timestamptz`	Optional, auto-updated

`guests` table

Column	Type	Source
`id`	`uuid` (PK)	GuestList.tsx
`wedding_id`	`text` (FK → websites.slug)	Links to website
`name`	`text`	Required
`email`	`text`	Optional

`attending`	`boolean`	Yes/No
`plus_ones`	`integer`	Number of +1s
`message`	`text`	Optional personal message
`created_at`	`timestamptz`	Auto-generated

> *[!NOTE] No SQL migration files exist in the repo. Schema is managed via Supabase dashboard.*

---

6. PAYMENT FLOW

Architecture

```
sequenceDiagram
    participant U as User (Browser)
    participant C as /api/checkout
    participant LS as LemonSqueezy API
    participant W as /api/webhooks/lemonsqueezy
    participant DB as Supabase (websites)

    U->>C: POST {slug} + Auth Bearer
    C->>C: Verify user owns site
    C->>LS: Create checkout session
    LS-->>C: Checkout URL
    C-->>U: {url: checkoutUrl}
    U->>LS: Redirected to payment page
    LS->>W: Webhook: order_created
    W->>W: Verify HMAC signature
    W->>DB: UPDATE status='production', payment_id
    LS-->>U: Redirect → /dashboard?payment=success&slug=X
    U->>U: ConfettiSuccess 🎉
```

Components

Component	File	Role
PublishButton	PublishButton.tsx	CTA that calls `/api/checkout`, redirects to LS. Has `compact` variant for editor. Shows "Publish — $49"
Checkout API	api/checkout/route.ts	Validates auth, verifies ownership, creates LS checkout with `website_slug` in custom data
Webhook	api/webhooks/lemonsqueezy/route.ts	On `order_created`, updates site `status` → `production` and stores `payment_id`
ConfettiSuccess	ConfettiSuccess.tsx	CSS confetti overlay + "Your site is live!" message. Auto-dismisses after 6s

Env Vars Required

```
LEMONSQUEEZY_API_KEY
LEMONSQUEEZY_STORE_ID
LEMONSQUEEZY_VARIANT_ID
LEMONSQUEEZY_WEBHOOK_SECRET
NEXT_PUBLIC_APP_URL
```

---

7. GUEST & RSVP SYSTEM

RSVP Modal

RSVPModal.tsx — Framer Motion animated modal that inserts directly into the `guests` table.

Data collected per guest:

- **Name** (required)
- **Email** (optional)
- **Attending** (Yes/No toggle)
- **Plus Ones** (counter, only shown if attending)
- **Message** (optional textarea)

GuestList Component

GuestList.tsx — Admin-facing table with:

- Summary stats: Total Responses, Attending, Not Attending, Total Party Size (including +1s)
- Full table with Name, Email, Status badge, Party Size, Message, Date

> *[!IMPORTANT] The `GuestList` component exists but is **not wired into any route**. There is no admin page to view guest RSVPs. It needs to be added to the dashboard or editor.*

---

8. DEMO vs PRODUCTION LOGIC

Demo Mode

Behavior	Implementation
Default state	Every new site starts as `status: 'demo'` (DB default)
Demo watermark	Yellow gradient banner at top: "⚡ This is a demo preview — Publish to go live" — demo/[slug]/page.tsx
Editor access	Shown for owners AND demo viewers (`showEditor = isOwner || isDemo`)
Dashboard badge	Yellow "Demo" pill

Production Unlock

1. User clicks "Publish $49" → LemonSqueezy checkout → webhook → `status = 'production'`
2. Dashboard badge → Green "Live" pill
3. "Publish" button replaced with "Share Link" (copies URL)
4. Demo watermark hidden (`isDemo = siteStatus !== 'production'`)

Public Access (Guests)

> *[!WARNING] **All sites are currently accessible without auth** at `/demo/{slug}`. Even production sites use the `/demo/` route. There is no separate `/site/{slug}` or custom domain route for production sites. The "Powered by Alt-Vows" footer is **not yet implemented**.*

---

9. KNOWN BUGS

#	Bug	Severity	Details
1	**Editor hardcoded to vintage-vinyl schema**	🔴 High	`EditorSidebar` imports `vintageVinylSchema` directly. Cannot edit The Voyager or future themes
2	**Save hardcodes `theme_id: 'vintage-vinyl'`**	🟡 Medium	All saves write `vintage-vinyl` regardless of actual theme
3	**Cyberpunk niche uses old JSON format**	🟡 Medium	`cyberpunk-wedding` in `niches.json` uses flat keys (`color`, `names`, `googleFont`) instead of nested `hero`/`theme` structure. Will crash `mergeConfig`
4	**ConfettiSuccess shows `/demo/` URL for live sites**	🟡 Medium	Production sites should have a clean URL, not `/demo/` prefix
5	**`GuestList` component not routed**	🟡 Medium	Built but never rendered — no way for site owners to view RSVPs
6	**No "Powered by Alt-Vows" footer**	🟡 Medium	The viral loop mechanic is missing
7	**RSVP Modal styling uses hardcoded classes**	🟠 Low	Uses `bg-cream`, `text-burgundy`, `text-warmBrown`, `text-gold` — these Tailwind v4 classes may not exist unless defined, could cause invisible text
8	**The Voyager disabled in dynamic routing**	🟠 Low	Commented out in `registry.ts`, so `/themes/` route can't render it
9	**Deep clone via `JSON.parse/stringify` in editor**	🟠 Low	Performance concern for complex theme objects with many nested keys
10	**Webhook signature skip on placeholder secret**	🟠 Low	Skips verification if secret is `'your_webhook_secret_here'` — must be updated before prod

---

10. WHAT IS NOT BUILT YET

Missing for MVP

Feature	Priority	Notes
"Powered by Alt-Vows" footer	🔴 P0	Core viral loop — must appear on every guest-facing site

Production URL route	🔴 P0	Sites need a clean `/s/{slug}` or custom domain route, separate from `/demo/`
RLS policies in Supabase	🔴 P0	Critical security — users can currently read/write any row
Guest list admin view	🟡 P1	Component built, needs route (e.g. `/dashboard/guests/{slug}`)
Dynamic editor schema	🟡 P1	Editor should load schema based on `theme_id`, not hardcode vintage-vinyl
Supabase image storage	🟡 P1	Currently using direct URLs / inline upload — no Supabase Storage bucket for user images
Email notifications	🟡 P1	RSVP notifications to site owners, confirmation to guests
SEO / OpenGraph per site	🟡 P1	No dynamic `<head>` metadata for shared links
Mobile responsiveness audit	🟡 P1	Themes are desktop-first, mobile polish needed
Password reset flow	🟡 P2	No forgot-password UI
Delete site	🟡 P2	No way to delete a created site
Site analytics	🟢 P3	Page view tracking for site owners
Custom domain support	🟢 P3	Future premium feature

Themes Planned but Not Built

Theme	Status
Gothic / Dark Romantic	Not started
Vintage / Boho	Not started
The Voyager (Travel)	Built but disabled in routing + editor

Deployment Status

> *[!CAUTION] **Not deployed.** No Vercel/hosting config found. The app runs locally only (`npm run dev`). No CI/CD pipeline.*

---

11. TECH STACK DETAILS

Full Dependency List

Package	Version	Role
`next`	16.1.6	Framework
`react` / `react-dom`	19.2.3	UI library
`@supabase/supabase-js`	2.95.3	Database + Auth

`zustand`	5.0.11	State management (editor)
`framer-motion`	12.29.2	Animations
`tailwindcss`	4.x	Styling
`@tailwindcss/postcss`	4.x	Tailwind integration
`three`	0.182.0	3D rendering (unused?)
`@react-three/fiber`	9.5.0	React Three.js bridge
`@react-three/drei`	10.7.7	Three.js helpers
`gsap`	3.14.2	Animation library
`lucide-react`	0.563.0	Icons
`clsx`	2.1.1	Class concatenation
`typescript`	5.x	Language
`eslint` + `eslint-config-next`	9.x / 16.1.6	Linting

Technical Debt & Architecture Concerns

Concern	Details
Three.js bundled but unused	`three`, `@react-three/fiber`, `@react-three/drei` = ~1.2MB added to bundle. Not referenced in any component. Remove if not planned
GSAP bundled but unused	`gsap` imported in `package.json` but no `gsap` imports found in source
Two different theme config patterns	Vintage Vinyl uses a custom `VinylThemeConfig` type + `mergeConfig()`. The Voyager uses the generic `ThemeConfig` interface with `schema[]`. Need to unify before adding more themes
No error boundaries	No React error boundaries anywhere — theme crashes will blank the whole page
No tests	Zero test files in the project
Server Components under-utilized	Most pages are `'use client'` — missing SSR/SEO benefits of RSC. The `/themes/[slug]` page is the only SSR page
`styled-jsx` usage	`ConfettiSuccess` uses `<style jsx>` — works in Next.js but inconsistent with Tailwind approach
No loading.tsx / error.tsx	No Next.js built-in loading or error pages

---

🎯 PRIORITIZED LAUNCH ROADMAP

Phase 1 — Security & Infrastructure (Week 1)

#	Task	Why
1	**Set up Supabase RLS policies**	🚨 Users can read/write other users' data
2	**Deploy to Vercel**	Need a staging URL for testing webhooks + sharing
3	**Configure LemonSqueezy webhook secret**	Currently set to placeholder value
4	**Remove unused deps** (three, gsap)	1.5MB+ bundle weight savings

Phase 2 — Core Missing Features (Week 2)

#	Task	Why
5	**Add production site route** (`/s/{slug}`)	Clean URLs for live sites, separate from editor/demo
6	**Add "Powered by Alt-Vows" footer**	Core viral loop = free marketing to 150 guests
7	**Wire up GuestList to dashboard**	Site owners need to see RSVPs
8	**Dynamic editor schema by theme_id**	Unblocks The Voyager + future themes
9	**Fix cyberpunk niche JSON format**	Currently broken/crashes `mergeConfig`

Phase 3 — Polish for Launch (Week 3)

#	Task	Why
10	**SEO / OpenGraph metadata per site**	Shared links need rich previews
11	**Mobile responsiveness pass**	Many guests will view on phone
12	**Enable The Voyager theme**	Second theme = more market coverage
13	**Add delete site + password reset**	Basic user management
14	**Supabase Storage for images**	User uploads need persistent storage

Phase 4 — Growth (Post-Launch)

#	Task
15	Email notifications (RSVP alerts)
16	Site analytics dashboard
17	Gothic / Boho themes
18	Custom domain support