

WHITE PAPER / CASE STUDY

Brand Spins

Architecting a brand-integrated digital music store for every screen — one commerce core, many channels

Product definition · Solution architecture · Multi-channel eCommerce · Data & marketing automation

By Daniel Brody, Schulich MBA

CTO, CTO Rescues

ctoescues.com

SEPTEMBER 2026



01 EXECUTIVE SUMMARY

One storefront. Every screen. Every brand.

BrandSpins set out to put a branded digital music store — “more music, more places” — in front of consumers wherever they are: on the desktop web, on phones and tablets, inside native apps, in downloadable players, and embedded in partners’ own websites.

The challenge was not building a single music store; it was building one commercial engine that could surface consistently across seven product lines, three mobile operating systems, a long tail of browsers and devices, and the account, payment and accounting systems of the brands that would offer the store to their own customers.

Daniel Brody defined the BrandSpins product and its solution architecture. The design centres on a **single storefront and eCommerce core** fed by the **MN Open** music-data API, surfaced through web, mobile, download-client and CMS-plugin channels, and integrated with brand partners through **single sign-on, payment-gateway and financial-system APIs**. Marketing automation and a **data warehouse** close the loop, turning every search, preview and purchase into reportable insight for BrandSpins and its partners.

PRODUCT FOCUS	MOBILE PLATFORMS	LOCALIZATION	CARDS STORED
7 → 4	3	3 × 3	0
product sections consolidated into four build workstreams	native apps for iOS, Android and Microsoft (Xbox 360 / Win8)	currencies (USD, CDN, Pound) × languages (EN, FR, ES)	no customer credit-card data held on the platform

Project snapshot

Client	BrandSpins, Inc. — La Jolla, California (tagline: “more music · more places”)
Client leadership	Billy Tuchscher, President & CEO
Sector	Digital music · branded / partner-integrated eCommerce
Engagement	Product definition, functional specification and solution / data-flow architecture
Author’s role	Daniel Brody - CTO / Chief Architect
Timeframe	Product and architecture documents dated September 2014

02 CLIENT & BACKGROUND

BrandSpins: more music, more places

BrandSpins, Inc. is a La Jolla, California company whose name and tagline describe the ambition: a digital music experience that brands can offer their audiences, delivered on every device those audiences use.

The BrandSpins product documents describe a music-commerce platform in which the storefront is presented in a **partner's context** rather than as a destination of its own. The store template carries the **customer's logo, menus and links back to the customer's site**, the footer carries the customer's branding alongside a "Powered by" platform credit, and the platform integrates with each partner's **consumer accounts (single sign-on), payment gateway and reporting**. In other words, BrandSpins was designed as a business-to-business-to-consumer (B2B2C) platform: BrandSpins serves brands, and brands serve music to their consumers.

Consumers, meanwhile, were expected to search, browse, preview, buy and download music — songs and albums — and play it back on desktop and mobile. Catalog content, artist information and customer identities were to be sourced through the **MN Open** music-data API, so that BrandSpins could focus on experience, commerce, partner integration and data rather than on content licensing infrastructure.

What BrandSpins needed from its technology leadership

- A clear, buildable definition of the product — which apps, stores, clients and plug-ins, and how they relate.
- A single architecture that keeps cart, account and purchase behaviour identical across every channel.
- A repeatable way to plug the store into each brand partner's identity, payment and accounting systems.
- A data and marketing foundation that turns consumer activity into insight and targeted engagement.

03 THE CHALLENGE

Many screens, many partners, one experience

The product documents make the difficulty explicit. The same commercial experience had to work everywhere, integrate with systems BrandSpins did not own, and stay compliant and secure along the way.

1. A fragmented device and browser landscape

The player and storefront had to be **cross-platform, customizable and instrumented for analytics**. The documented support targets spanned modern and legacy environments alike: Chrome, Firefox, Internet Explorer, Safari and Opera on Windows (including legacy **IE6 through IE11**); Safari, Firefox, Chrome and Opera on OSX; Mobile Safari on iPad, iPhone and iPod Touch; the **Android 2.3** browser; and the **BlackBerry OS 7** phone and PlayBook browsers. As the design notes put it, layouts had to be responsive “because [there are] too many variations of device screens.”

2. Seven product lines that must behave as one

BrandSpins was defined as a family of products — desktop and mobile web stores, OS-independent download clients, an OSX/Windows music player, native mobile store and player apps, marketing automation, data analytics and CMS plug-ins. Without a shared core, each would drift into its own cart, its own accounts and its own data. The native mobile app, in particular, was specified as a true app (“not [a] mobile web page running from the store”) that nonetheless had to share **the same cart functions** as the eCommerce site.

3. Integration into partners' systems

Every brand partner brings its own consumer accounts, payment gateway and reporting needs, and BrandSpins had to integrate its own store and payment gateway into its **internal accounting system**. The platform therefore needed clean, API-based integration points for single sign-on, payments, financial systems and client reports.

4. Licensed content delivered securely

Catalog data, search, previews and downloads depend on a third-party music-data service (MN Open). The architecture had to broker **secure requests for data** from each client and return content and metadata reliably across channels — while customer IDs were created and validated against MN Open.

5. Compliance, trust and an undefined marketing layer

Email had to be split into **transactional** and **marketing** streams for spam compliance, with explicit consumer opt-in for newsletters, partner information and promotions. Payments had to be handled

without **holding customer credit cards** on the platform. And an early working draft flagged marketing automation itself as an open question — “define this? What is this? What does it do?” — which had to be answered before it could be built into each product.

04 PROJECT SCOPE & OBJECTIVES

From seven product sections to four workstreams

The engagement first catalogued the full product surface, then grouped it into workstreams that a delivery organization could own. The initial definition identified seven product sections:

#	Product section	Scope detail (from the product definition)
1	Desktop web-based music store	Primary storefront for search, browse, preview, purchase and download
2	Web-based mobile music store	Phone and tablet size formats
3	Download clients & desktop player	OS-independent mobile download client OS-independent desktop download client OSX and Windows based music player
4	Mobile apps	Mobile music store app and mobile music player — iOS, Android and MS (Xbox 360 / Win8)
5	Marketing automation	Mail, social and consumer-profile tracking
6	Data analytics & reporting	Data warehouse
7	API plug-ins & integration	API plug-ins for WordPress and other CMS solutions Customer integration: SSO with client consumer accounts, client payment gateway, client reports Financials integration: payment gateway and store into internal accounting

Objectives

- **One commerce core** — a single storefront/eCommerce engine behind every channel, so cart, orders, accounts and coupons behave identically everywhere.
- **Every screen** — responsive web for desktop, phone and tablet; native apps on the three major mobile platforms; download clients and a desktop player.
- **Plug into partners** — standard APIs for single sign-on, payment gateways, financial systems, client reports and CMS embedding.
- **Compliant by design** — no stored card data; separated transactional and marketing email with explicit opt-in.
- **Data as a product** — a data warehouse and reporting suite serving both BrandSpins operations and brand partners.

- **Ready for multiple markets** — multi-currency and multi-language from the start.

These sections were then consolidated into **four product-development workstreams**: Web (desktop and mobile), Mobile Application, Marketing & Data Analytics, and Third-Party Connections.

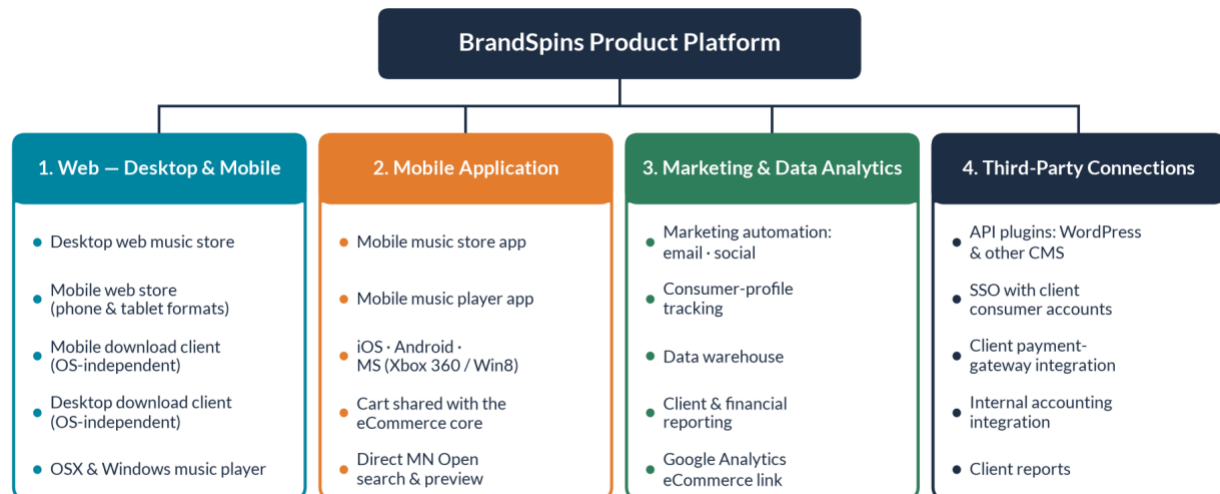


Figure 1 — BrandSpins product platform: the four product-development workstreams (redrawn from the BrandSpins product definition).

05 SOLUTION & ARCHITECTURE

A hub-and-spoke platform around a single storefront core

The BrandSpins architecture places one storefront and eCommerce data hub at the centre. Content and identity flow in; every consumer channel, partner system and analytics service connects to it through defined interfaces.

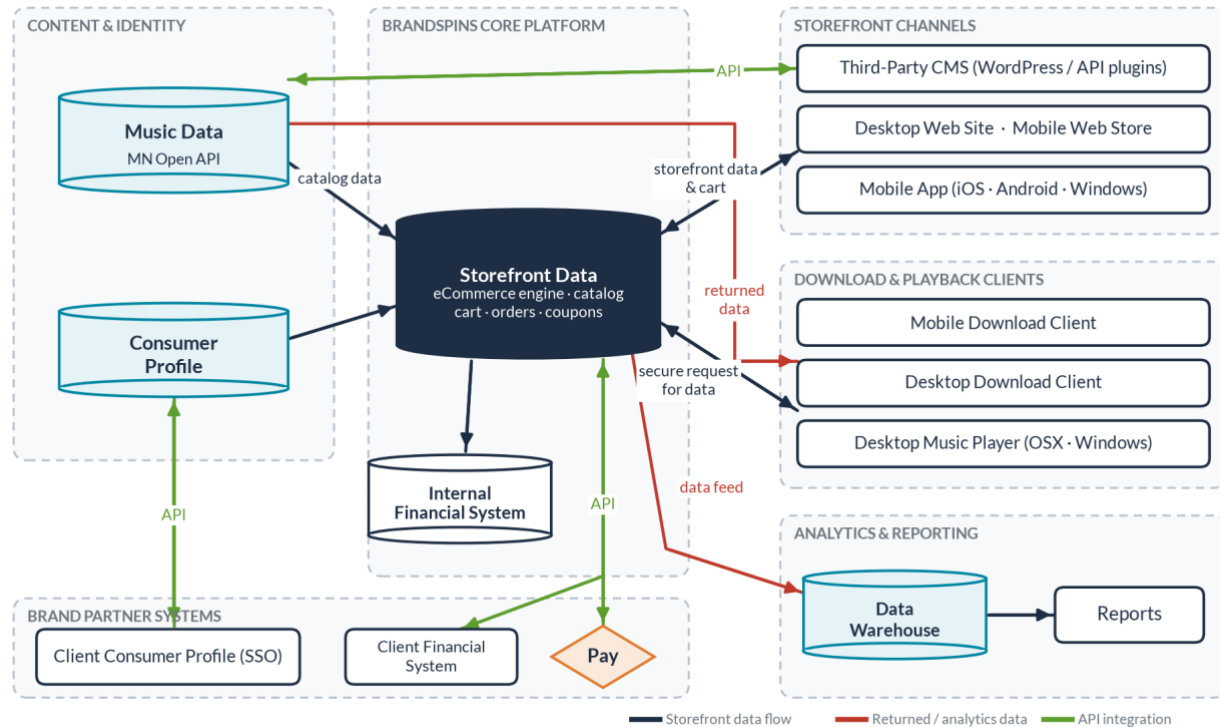


Figure 2 — BrandSpins data-flow architecture (clean redraw of the original design diagram; see Appendix A for the original).

How the data flows

- **Content & identity in.** Music data from the MN Open API feeds the storefront catalog. The consumer profile feeds the storefront and synchronises, via API, with each brand partner's client consumer profile to enable single sign-on.
- **Storefront channels.** The desktop web site, mobile web store and native mobile app exchange storefront data (catalog, cart, account) with the core, so the cart behaves the same on every channel. Third-party CMS plug-ins (e.g., WordPress) connect via API to the music data service.
- **Download & playback clients.** Mobile and desktop download clients and the desktop music player make a **secure request for data** through the storefront; the music data service returns the requested content and data directly to the client.
- **Money.** The storefront connects via API to the payment gateway ("Pay") and to the client financial system, and posts to BrandSpins' internal financial system — without holding card data.
- **Insight.** Storefront data flows into a data warehouse that produces operational, partner and financial reports.

Solution pillars

1. Unified storefront & eCommerce core

The catalog is supplied through API integration and organised by genre, product, album (a group of songs), song, artist, downloads and reviews, with global attributes such as label, release date, format (MP3), running time and MNET ID.

API-fed enrichment adds factual artist bios, links to artists' tweets and Facebook status updates, album abstracts, concert/event tickets and news, band members, similar artists and external links.

Sales management covers orders (order ID, invoice ID, customer, totals, status), with song purchases fixed at one unit, customer validation through the MN Open API, and payments through a payment-system API integration.

2. Cross-platform player built on proven open source

Rather than building a player from scratch, the design adopted **jPlayer**, a free, MIT-licensed JavaScript media library (a jQuery and Zepto plug-in) that the documentation notes was used by Pandora, the BBC and WNYC. Version 2.6.0 had been released in April 2014.

jPlayer renders with HTML5 where available and falls back to Flash, covering mp3, mp4 (AAC/H.264), ogg, webm and wav in HTML5 and mp3, mp4, rtmp and flv in Flash — and it was specified for integration into the native apps as well as the web store.

Three player patterns were evaluated: a standard audio player, a compact circle player suited to samples, and a playlist player (Figure 3).

3. Native mobile apps connected to the same core

The mobile apps were specified as native programs for iOS, Android and Windows Mobile — not a mobile web page — with UI designed for the wide range of screen sizes and resolutions across the three platforms.

The apps connect to the eCommerce system so **cart functions are identical** to the web store, and connect directly to MN Open for search and preview.

4. Brand-partner integration & white-label presentation

Standard integration points for **single sign-on** with client consumer accounts, the **client payment gateway**, **client reports**, and integration of the payment gateway and store into the internal accounting system.

A templated storefront that carries each customer's logo, menus and site links in the header, and customer branding plus a "Powered by" credit in the footer; API plug-ins bring the store into WordPress and other CMS platforms.

5. Customer accounts & marketing automation

Customer profiles (created via the MN Open API) capture name, contact details and password, plus marketing attributes — date of birth, gender, social connections, newsletter and partner opt-ins — and Facebook connectivity to retrieve a profile, post a link or upload a photo.

Account pages provide purchase history, a download manager and a sample player.

Marketing automation was defined to deliver **targeted marketing email**, a **dynamic consumer profile** that feeds back consumer interactions, and a **social-media tool** for tracking, response and posting from a central location. Email is split into transactional and opt-in marketing streams, and a full

coupon engine supports percentage or fixed discounts, minimum totals, start/end dates, per-coupon and per-customer usage limits, and product targeting.

6. Data analytics, reporting & system administration

A data warehouse supports customer invoices, sales reports, products viewed and purchased, customer demographics, cart abandonment, and end-of-day, end-of-month and end-of-year financial reports, with a link to Google Analytics for eCommerce performance reporting.

System settings include admin users, multi-currency (USD, CDN, Pound), multi-language (English, French, Spanish), error logs and backup/restore.

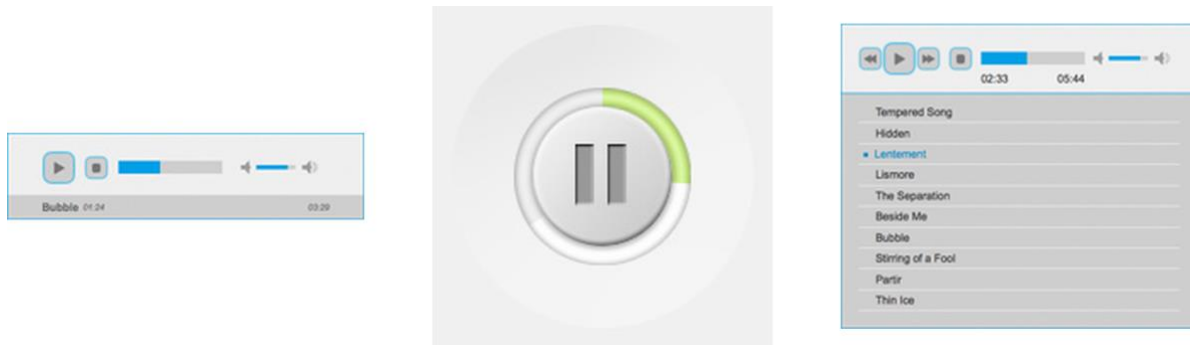


Figure 3 — jPlayer interface patterns evaluated for BrandSpins: audio player, circle player for samples, and playlist player (from the product specification).

Discovery-led storefront design

The storefront design was specified to work in both mobile and desktop environments across all platforms, using responsive blocks and columns. Template elements include music search (via the MN Open API) with results filtered by how the consumer searched and “more by this artist”; browsing by all genres, a specific genre, artist, album or song; music discovery through content recommendations; **trending** modules for today’s popular artist, album and song; and data blocks featuring a music sample player.

06 TECHNOLOGY STACK

Build where it differentiates, integrate where it doesn't

The documented stack favoured proven third-party services and open-source components for commodity capabilities, reserving custom build for the commerce core, partner integrations and data layer.

Layer	Technology / approach	Role in the platform
Music content & metadata	MN Open API	Catalog, search, preview, downloads; customer ID creation and validation
Commerce core	Storefront / eCommerce engine	Catalog, cart, orders, invoices, customers, coupons, reports
Web experience	Responsive web (desktop, phone, tablet)	Block/column layouts adapting to the variety of device screens
Media playback	jPlayer (MIT) · jQuery / Zepto · HTML5 with Flash fallback	Cross-browser audio/video player, sample and playlist players
Mobile	Native apps — iOS, Android, Microsoft (Xbox 360 / Win8)	Store and player apps sharing the eCommerce cart
Desktop clients	OS-independent download clients · OSX/Windows player	Secure download and playback outside the browser
Payments	Payment-gateway API (incl. client gateways)	Payment processing without storing card data
Identity	SSO integration with client consumer accounts · Facebook connect	Seamless partner log-in; social profile and sharing
Financials	Integration with internal and client financial / accounting systems	Revenue posting and reconciliation
Data & analytics	Data warehouse · Google Analytics eCommerce	Sales, product, demographic, abandonment and financial reporting
Distribution	API plug-ins for WordPress and other CMS	Embedding the store in partners' own sites

07 IMPLEMENTATION APPROACH

From product definition to buildable blueprint

The engagement followed a definition-first approach: establish what the product is, organise it for delivery, specify it in detail, and make the architecture explicit before build.

1. **Product definition** — catalogued the full BrandSpins product surface into seven product sections spanning web, mobile, desktop clients, marketing, data and integration.
2. **Workstream consolidation** — regrouped the seven sections into four product-development workstreams (Web desktop & mobile, Mobile application, Marketing & data analytics, Third-party connections) that teams could own end-to-end.
3. **Functional specification** — detailed the player, eCommerce (catalog, sales, customers, marketing, design, system, reports) and mobile app (layout, cart, player, account manager) requirements.
4. **Open questions surfaced and resolved** — working drafts explicitly flagged undefined areas, such as marketing automation, and answered them before they were folded into each product.
5. **Data-flow architecture** — mapped every system, channel and partner integration onto a single data-flow diagram (Figure 2 and Appendix A), maintained as a living document.
6. **Build-vs-buy decisions** — adopted MN Open for content, jPlayer for playback and payment-gateway APIs for payments, focusing custom work on the commerce core and integrations.

Delivery details for Daniel to complete

[PLACEHOLDER: build team / vendors, delivery phases and timeline]

[PLACEHOLDER: which channels and workstreams went live, and when]

08 RESULTS & BUSINESS IMPACT

What the architecture delivered

The source material for this case study documents the product definition and architecture rather than post-launch performance, so this section distinguishes what the engagement **delivered** from the **business outcomes the design was built to enable**. Quantified results are left as clearly marked placeholders.

Engagement deliverables

- A complete **product definition** covering seven product sections and four delivery workstreams.
- A detailed **functional specification** for the player, eCommerce platform and mobile apps.

- An end-to-end **data-flow architecture** connecting content, identity, channels, payments, financial systems and analytics.
- A documented **player technology evaluation** and build-vs-buy direction for content, playback and payments.
- A **reusable blueprint**: the same product structure was subsequently adapted into a “Fntsy Games Product” specification (January 2015).

Business outcomes the design enables

Outcome	How the architecture supports it
Faster channel expansion	New channels (a new app, client or CMS plug-in) connect to an existing storefront core instead of re-implementing commerce.
Consistent customer experience	One cart, one account and one purchase history across web, mobile app and clients.
Partner-ready onboarding	Standard SSO, payment, financial and reporting interfaces plus a branded template make each new brand partner a configuration exercise rather than a rebuild.
Reduced compliance exposure	No card data stored on the platform; spam-compliant separation of transactional and opt-in marketing email.
Monetizable insight	A data warehouse and reporting suite give BrandSpins and its partners visibility into sales, products, demographics and cart abandonment.
International reach	Multi-currency and multi-language support from the outset.

09 LESSONS LEARNED & KEY TAKEAWAYS

What still applies in 2026

Although BrandSpins was specified in 2014, its core decisions map directly onto how multi-channel platforms are built today — API-first, headless and composable.

01 Design the core once; let channels multiply.

A single storefront/eCommerce hub with API-connected channels is what allowed seven product lines to share one cart and one customer record. Today's API-first, headless commerce follows the same principle.

02 Treat partners as first-class integrations.

In a B2B2C model, SSO, payment, financial and reporting interfaces are the product. Defining them up front turns partner onboarding into configuration rather than custom work.

03 Buy the commodity, build the differentiator.

Licensed content (MN Open), open-source playback (jPlayer) and payment gateways let the team focus engineering on commerce, integration and data — where BrandSpins could differentiate.

04 Keep sensitive data out of scope.

Not holding card data, and separating transactional from opt-in marketing email, reduces compliance burden and builds consumer trust from day one.

05 Make data a product, not an afterthought.

Designing the data warehouse, reports and marketing automation alongside the store ensures every interaction is captured for both the operator and its partners.

06 Name the unknowns early.

Flagging undefined areas such as marketing automation in working drafts — and forcing a definition — prevents ambiguity from leaking into the build.

Why it matters

BrandSpins shows how disciplined product definition and architecture de-risk an ambitious multi-channel platform before code is written. By anchoring every channel and partner to one commerce and data core, the design converts a sprawling product vision into a scalable, partner-ready business — the same outcome-first approach CTO Rescues brings to every engagement.

One commerce core · Every screen · Every brand

10 ABOUT THE AUTHOR & CTO RESCUES

Daniel Brody, Schulich MBA — CTO, CTO Rescues

Daniel Brody is an enterprise CTO/CIO with 30+ years of global experience helping companies stabilize, scale and transform mission-critical technology platforms. He has led technology strategy and execution across regulated industries including financial services, healthcare, gaming and enterprise SaaS, and holds multiple patents in digital systems and data architecture.

His MBA brings financial discipline to every technical decision: engagements are framed around EBITDA improvement, risk reduction and capital-event readiness, not just engineering quality. Daniel works directly with CEOs, boards and investors as a trusted advisor and operating partner.

CTO Rescues is the practice around Daniel: a fractional CTO / CIO-for-hire bench assembled for each engagement — **design-it · sell-it · build-it · implement-it**. The work takes startups from MVP to market launch, prepares scaleups for an IPO and beyond, and helps small and midsize businesses modernize, grow, cut cost and raise shareholder value. Its portfolio spans carbon registries, healthtech, iGaming, fintech, marketing technology, enterprise ERP and digital media.

Expertise

Product & IP commercialization

Taking technology from architecture to market — roadmap, build leadership, GTM alignment and investor documentation.

M&A, exit & due diligence

Pre-transaction platform hardening, technical due diligence, post-merger integration and CTO transition support.

Platform turnaround

Rapid stabilization for platforms under stress and restored delivery velocity.

Cloud & AI transformation

Cloud modernization across AWS, Azure and GCP; AI-enabled ERP and enterprise data architecture.

READY TO BUILD, SCALE, OR EXIT?

Let's discuss your technology challenge.

If your challenge is product velocity, platform reliability, multi-channel architecture, technical due diligence or exit preparation, book a Fractional CTO call or reach out directly.

Book a Fractional CTO Call → ctoescues.com/contact

Email → dan@ctoescues.com

More case studies → ctoescues.com/casestudy

11 APPENDIX A

Original BrandSpins data-flow diagram

The original data-flow diagram from the BrandSpins product documentation, reproduced for reference. The redrawn version in Figure 2 preserves its systems and flows.

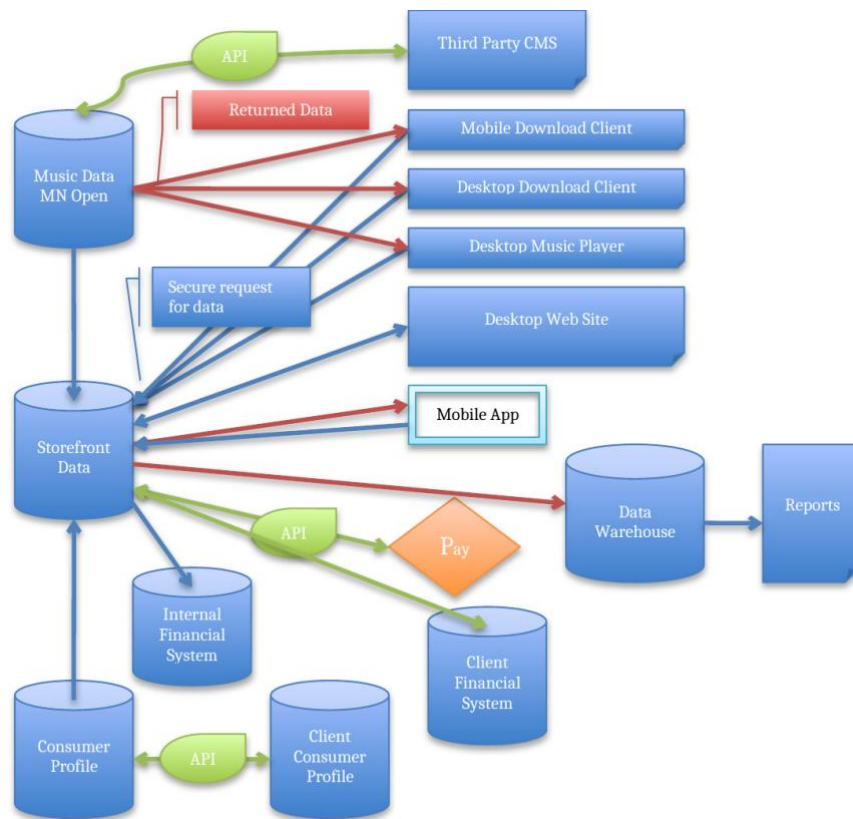


Figure 4 — Original BrandSpins data-flow diagram (product documentation, 2014). Red: returned data; blue: storefront data and secure requests; green: API integrations.

Appendix B — Documented platform & media support

Platform	Browsers / environments
Windows	Chrome, Firefox, Internet Explorer, Safari, Opera; legacy IE6, IE7, IE8, IE9, IE10, IE11
OSX	Safari, Firefox, Chrome, Opera
iOS	Mobile Safari — iPad, iPhone, iPod Touch
Android	Android 2.3 browser
BlackBerry	OS 7 phone browser, PlayBook browser
Media — HTML5	mp3, mp4 (AAC/H.264), ogg (Vorbis/Theora), webm (Vorbis/VP8), wav
Media — Flash	mp3, mp4 (AAC/H.264), rtmp, flv