CLEVYR

HubSpot Custom Integration Accreditation

Item One: Documented Integration Plan

T-Mobile RCC Captioning Ordering Platform — Drupal to HubSpot Migration

Submission Table of Contents

Section 1: Summary of Customer Needs

Customer: T-Mobile Accessibility (https://www.tmobileaccess.com/)

Portal ID: 242363277

Private App: Relay-Theme

Project: Migration of the Relay Conference Captioning (RCC) Ordering Platform from a Drupal 10 site to HubSpot CMS Enterprise, with a custom integration to Verbit's "One Order" API.

Challenges / Pain Points:

- The legacy Drupal ordering system was dated, rigid, and presented users with a single long-form page rather than a guided, accessible flow.
- State-specific configuration (logos, disclaimers, time zones, BCC recipients, zip codes) was managed manually across individual forms/code rather than from a single location, creating

ongoing maintenance burden for T-Mobile's internal team and requiring developer involvement for routine changes.

- HubSpot's native form components could not fully satisfy WCAG 2.2 AA accessibility requirements out of the box, which T-Mobile's Accessibility Review Center (ARC) requires for this platform.
- The prior Vitac API integration was being deprecated in favor of Verbit's new "One Order" API, which introduced new capabilities (recurring meetings, ASR captioning, post-production requests) that the legacy system could not support.
- Limited visibility into submission volume and service-type trends; no built-in reporting existed in the Drupal system.

Systems Integrated with HubSpot

- **Verbit "One Order" API** (orders.verbit.co) — human and ASR captioning, recurring meetings, post-production requests.

Other Systems Leveraged

- No ETL tooling was used, and no historical Drupal submission data was migrated into HubSpot — the new system launched with a clean data set. Elevated Third (the outgoing Drupal vendor) provided SFTP and architecture access for reference during discovery.
- State-specific content logic that existed on the Drupal site — namely which attestation content/checkboxes each state's form required — was carried forward to preserve legal/compliance parity. This wasn't a data migration in the technical sense; it was a content-parity exercise, using a reference spreadsheet to verify each state's required content against what existed on the legacy Drupal site before building it into the new HubDB-driven form logic.

Section 2: High-Level Outline of Proposed Solution

2.1 Data Sources and Targets

Source	Target	Data
User-submitted captioning request (custom ARIA-compliant form)	HubSpot Contact record	Requester name, email, and related submission details
Custom form submission	HubSpot Ticket (Service Request) record	Meeting details, state, service type
HubSpot Ticket record	Verbit "One Order" API (orders.verbit.co/api/v2/orders)	Meeting request payload for live/recurring booking
Verbit "One Order" API response	HubSpot Ticket record	Caption URL, Caption ID, booking confirmation

Source	Target	Data
HubDB (State Forms table)	Custom form module + notification workflows	State name, zip codes, disclaimers, time zones, BCC email addresses, approval requirements

2.2 Visual Diagram

See the System Overview diagram and Verbit Integration Overview diagram (originally built as a Mermaid flow) included with this submission — these illustrate the end-to-end flow from form submission through Contact/Ticket creation, Verbit API authentication and order posting, and outbound notifications. These same diagrams are strong candidates to carry forward into Item 2, with authentication method and Portal ID/app details layered on.

Integration Diagram - Item Two: Caption Relay (Claude Asset)

2.3 Integration Solution Description

The integration operated through four chained HubSpot workflows:

1. **Workflow #1** authenticated with the Verbit API and retrieved a Bearer Token.
2. **Workflow #2** posted the meeting request to Verbit's API endpoint (orders.verbit.co/api/v2/orders) to create a live or recurring booking.
3. The system updated the Ticket record with the Caption URL and Caption ID returned from Verbit.
4. **Workflow #3** sent a confirmation email to the requester with meeting and caption details.
5. **Workflow #4** queried HubDB to identify the corresponding state configuration and dispatched notifications to designated state recipients.

For states requiring approval (e.g., Texas), an additional approval stage was inserted between Ticket creation and the Verbit API call: the Ticket entered a "Requested" stage, approvers (managed via a HubDB-driven access list and restricted portal access) reviewed the request, and only on approval did the Ticket move to "Approved" and trigger the Verbit API call. Denied requests moved to a "Denied" stage with an automated requester notification.

Authentication between HubSpot and Verbit used a token-based handshake, with tokens rotated per T-Mobile's internal security policy and transmitted over HTTPS.

2.4 Why This Integration Mechanism Was Chosen

Three alternatives were considered and ruled out:

- **Marketplace/native connector:** No existing HubSpot marketplace integration supported Verbit's "One Order" API, and no native connector could satisfy the bidirectional, state-conditional logic (approval routing, HubDB-driven notification routing) this project required.

- **Zapier:** Ruled out per the accreditation's own disqualifying criteria, and impractical regardless given the volume of conditional logic (per-state approval routing, dynamic BCC lists) needed at each step.
- **Operations Hub alone:** Considered for the workflow/automation layer, but Operations Hub's native form components could not meet WCAG 2.2 AA accessibility requirements, which was a non-negotiable constraint from T-Mobile's Accessibility Review Center. A custom-built ARIA-compliant form module was required regardless of which automation layer sat behind it.
- **Custom development (selected):** Custom HubSpot workflows combined with a custom-built ARIA-compliant form module and serverless calls to the Verbit API were the only approach that satisfied all three hard requirements simultaneously: full accessibility compliance, bidirectional multi-object data flow (Contact + Ticket + HubDB), and state-specific conditional logic driven from a single, non-technical, editable source of truth.

Operations Hub was formally evaluated as part of this comparison. The team's early assessment pointed toward custom development based on the accessibility and conditional-logic requirements, and a follow-up research phase was conducted specifically to pressure-test that assumption before committing — confirming that Operations Hub's native form and workflow capabilities could not meet WCAG 2.2 AA requirements without significant custom work of its own, at which point custom development was formally selected.

2.5 Customer Requirements Not Included

- **Multi-step form flow:** A progressive, multi-step form was scoped as the preferred UX improvement over the legacy single-page form, and shipped as the multi-step version as part of the December 16, 2025 launch. The single-step fallback was documented as a contingency in case multi-step proved to be an accessibility barrier during ARC review, but that scenario never materialized — the multi-step form was the only version built and deployed.
- **Formal approval workflow for most states:** Per direct guidance from T-Mobile (Barbara Garcia), no formal approval process was implemented for the majority of states — only notification at time of submission. Texas was the sole exception, per state-specific administrative requirements. This was a deliberate scope decision, not an oversight.
- **Historical data migration:** Confirmed — no migration of legacy Drupal submission history into HubSpot was included in scope; the new system launched with a clean data set. State-specific attestation content was preserved via a content-parity review (see Section 1), which is distinct from a data migration.

Section 3: Deployment / Rollout Plan

3.1 Timeline

Milestone	Task	Target Date	Owner(s)
Kickoff	Project kickoff with T-Mobile	Kickoff Monday	Ashley Quintana, Jessica Ichien (Clevyr)
Milestone 1	Discovery & Planning	Week of Oct 13, 2025	Ashley Quintana, Jessica Ichien
Milestone 2	Platform Migration	Week of Oct 20, 2025	Bryan Patrick, Clevyr build team
Milestone 3	User Interface Development	Week of Nov 3, 2025	Bryan Patrick (Front-end Dev)
Milestone 4	Meeting Integration & API Development	Week of Nov 17, 2025	Aaron Krauss (Back-end Dev)
Milestone 5	Compliance & Data Management	Week of Dec 1, 2025	Jer Brand, Peter Evans
Milestone 6	Advanced Workflows & Validation	Week of Dec 15, 2025	Peter Evans, David Welch
–	Anticipated Site Launch	December 18, 2025	Clevyr build team
Milestone 7	Launch & Post-Launch Support	Week of March 15, 2026	Clevyr
Milestone 8	Accessibility Audit & Compliance	Week of Jan 12, 2026	Jer Brand (Accessibility Engineer)

Actual outcome: the platform launched ahead of the anticipated date — cutover began the evening of December 15, 2025 and the live site was confirmed fully operational in the early morning hours of December 16, 2025.

Dependencies ran sequentially: HubDB schema/state configuration had to be finalized before form module development; the form module and Verbit API integration both had to be functionally complete before compliance/accessibility validation and final QA; and T-Mobile stakeholder sign-off (Barbara Garcia) was the final gate before cutover.

3.2 Testing Plan

- **Accessibility testing:** Jer Brand conducted WCAG 2.2 AA compliance testing, including screen reader compatibility, keyboard navigation, and validation summary/error handling review, ahead of T-Mobile's own ARC review.
- **Integration/QA testing:** Peter Evans validated Drupal-to-HubSpot data mapping and tested the Verbit API integration end-to-end, including the approval-required flow for Texas.
- **Stakeholder UAT:** Confirmed — T-Mobile (Barbara Garcia) reviewed the live form flow and state-specific behavior prior to sign-off.

3.3 Deployment Schedule

- **Cutover approach:** occurred as a single cutover across all states, rather than a phased or state-by-state rollout.

- Anticipated launch per the original project plan was December 18, 2025; the platform actually launched ahead of schedule, with cutover beginning the evening of December 15, 2025 and the site fully live and confirmed operational in the early morning of December 16, 2025.

3.4 Post-Deployment & Maintenance Plan

- State-specific configuration (logos, disclaimers, time zones, BCC recipients, approval requirements) is maintained directly by T-Mobile's internal team via the HubDB State Forms table — no developer involvement required for routine state changes.
- Captioning request data is retained for 30 days post-submission, after which a custom-coded workflow automatically deletes the associated Ticket record; Contact data (name, email) is retained for ongoing reference and reporting.
- A formal support retainer and SLA are in place with Clevyr for post-launch support. Clevyr's project management team currently serves as the primary point of contact for ongoing issues; ownership is expected to transition to another Clevyr team member later in 2026 following completion of a planned online-cancellations feature enhancement.

Section 4: Error Handling / Logging Process

If an error occurs at any point in the workflow chain — including the Verbit API authentication call (Workflow #1) or the order-posting call (Workflow #2) — the following methodology applies:

1. **Logging:** The error is logged to a dedicated error property on the Service Request (Ticket) record, capturing the details of what failed.
2. **Stage transition:** The Service Request moves to a distinct "Error" stage in the pipeline, separate from Requested, Approved, Scheduled, Denied, and Cancelled. This makes failed requests immediately visible and distinguishable from requests that are simply pending action.
3. **Alerting:** The lead developer is alerted directly when an authentication failure occurs. More broadly, the development team receives an email notification any time an error occurs on a Service Request, ensuring failures are surfaced in real time rather than relying solely on someone noticing a stalled Ticket.
4. **Resolution:** A Service Request in the Error stage requires manual correction; it must be manually reset before it can continue moving through the remainder of the workflow.

This gives the team both immediate notification (email alert) and a persistent, auditable record (the error property and Error-stage Ticket) of every failure, satisfying the need for a clear, repeatable error-management methodology rather than relying on pipeline stage-watching alone.