Foil — eBay API Compliance

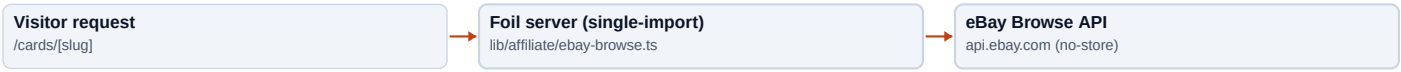
Last reviewed 2026-05-24 · Commit a8208dc · https://foiltcg.com/legal/ebay-api-compliance

What Foil does with eBay API data

This page documents how Foil complies with eBay's 2025 License Agreement, Buy APIs program terms, Marketplace Account Deletion compliance, and the eBay Partner Network agreement. It is the public mirror of Foil's internal compliance documentation. Every requirement listed below is enforced by code that is structurally tested in continuous integration — a regression on any one of these requirements fails the build before it can reach production.

Architecture

Foil's eBay integration follows a deliberately narrow architecture. Server-side code calls the Browse API at request time, reads the response, renders the page, and discards the response. There is no caching layer, no listing database, no AI training pipeline that touches eBay data.



Response discarded after render. Deletion webhook at /api/webhooks/ebay-marketplace-deletion handles eBay's HMAC challenge synchronously and persists nothing.

- **Render-time only.** Listings fetched server-side on each page request, response discarded the moment the page renders.
- **No persist.** No cached_listings table; no listing payload columns anywhere in the schema; telemetry rows store operational metadata only.
- **No train.** Content-generation pipeline never imports lib/affiliate/* and never calls api.ebay.com — architectural absence, not a runtime check.
- **CI-enforced.** Six structural invariants in lib/__tests__/ebay-compliance-invariants.test.ts fail the build on any regression.

Compliance requirements

#	Requirement	How Foil complies
1	**2025 License Agreement — no caching of listing payloads.**	Every eBay Browse API response is fetched server-side at the moment a visitor loads the page and discarded the moment the response returns. The Next.js route is force-dynamic; the underlying fetch uses cache: 'no-store'. Foil persists no listing data in any cache or database — no cached_listings table exists in the schema, and the function signature that would let a future contributor add one has no caching parameter.
2	**2025 License Agreement — no AI training on eBay data.**	Foil's content-generation pipeline (autonomous blog posts) never imports the affiliate / Browse module and never calls api.ebay.com. The architectural absence is enforced by a structural test that fails the build if the boundary is crossed. Listing data is read render-time and discarded; nothing from it ever reaches a model prompt or training corpus.
3	**2025 License Agreement — no AI-generated claims about specific listings.**	Editorial copy on per-card landing pages (the 'About this card' block) describes the card itself — its set, release year, rarity, and history. AI-generated copy never references specific live listings, prices, sellers, or item IDs. The 'Best current listing' block on each page self-describes from the Browse response at render time and changes on every page load.
4	**Marketplace Account Deletion — public webhook required.**	Foil subscribes to eBay's Marketplace Account Deletion notifications via a public webhook at https://foiltcg.com/api/webhooks/ebay-marketplace-deletion. The endpoint handles eBay's GET verification challenge (SHA-256 of challenge + verification token + endpoint URL) and HMAC-SHA256 verifies every POST notification. The production keyset is verified compliant in the eBay developer dashboard.
5	**Marketplace Account Deletion — handler must respond within 3 seconds, must not persist user data.**	The webhook handler is synchronous — no database writes, no outbound fetches, no awaited work beyond reading the request body. Every POST acknowledges with HTTP 200 and discards the payload. The endpoint stores nothing about eBay users, by design.
6	**Browse API auth — `client_credentials` grant with application scope only.**	Foil authenticates against eBay's Browse API using OAuth's client_credentials grant. The only scope requested is the public Browse scope (https://api.ebay.com/oauth/api_scope) — no user-context scopes, no sell.* scopes, no commerce.* scopes, no fulfillment scopes. Tokens are short-lived (2-hour TTL) and held in process memory only.
7	**Browse-call telemetry — operational metadata only, no listing payload.**	Foil maintains an internal telemetry table that records when Browse calls happened, which surface initiated each (page render vs alert cron), whether the call succeeded, and how long it took. The schema has exactly five columns: id, called_at, surface, success, latency_ms. There are no columns for titles, prices, item URLs, card slugs, sellers, or any other listing-payload field. The telemetry API signature has no parameter that could pass listing data through.
8	**EPN affiliate attribution — every outbound eBay click stamped with campaign + custom id.**	Every outbound eBay link Foil generates carries the documented EPN tracking parameters (mkevt, mkcid, mkrid, toolid) plus Foil's campaign ID and a custom-id identifying the originating surface (foil-card-page for the per-card page, foil-wishlist-alert for the alert cron). Affiliate URL construction is centralized in a single module; a structural test fails the build if any other code path constructs these parameters directly.
9	**Browse API rate limits — default 5,000 calls/day, must not exceed quota.**	The hourly wishlist alert cron caps itself at 200 Browse calls per run (200 × 24 = 4,800 per day, leaving 200 headroom for live page renders). A daily Discord summary surfaces 'approaching daily ceiling' when 24-hour usage crosses 80% of 5,000. Foil's Application Growth Check application is pending to lift the cap once organic traffic justifies it.
10	**Marketplace ID required on Buy API calls.**	Every Browse API call carries the X-EBAY-C-MARKETPLACE-ID: EBAY_US header. Foil serves the US marketplace exclusively at launch; the header is hard-coded rather than parameterized so a typo can't silently route requests to the wrong marketplace.
11	**Credentials must not be logged or surfaced to clients.**	OAuth credentials (App ID and Cert ID) are server-side environment variables. They are never logged, never returned in any response body, and never embedded in any client-side bundle. The OAuth access token itself is held in module-level memory only and never logged. The Browse module is server-only — it cannot be imported from any client component without a build error.
12	**No fallback to scraping eBay search HTML.**	Foil's only server-side connection to eBay is the official Browse API. The fallback CTA shown when the Browse response is empty is a navigable browser-facing eBay search URL (sponsored via EPN affiliate parameters) — but it is a link the visitor's browser follows, not a server-side fetch. The 2025 License Agreement's prohibition on automated HTML scraping is honored architecturally: Foil constructs no fetch call to www.ebay.com from server-side code.

Every row has a CI guard; details at https://foiltcg.com/legal/ebay-api-compliance and on request.

Marketplace Account Deletion

Marketplace Account Deletion notifications are received at a public webhook (https://foiltcg.com/api/webhooks/ebay-marketplace-deletion). The handler verifies eBay's HMAC signature, acknowledges with HTTP 200, and discards the payload. Foil persists no eBay user data of any kind.