ChatGPT

Payment Systems as Cognitive Infrastructure

Noel Michael Conrad
OpenAI
21 June 2026

Abstract and Design Premises

This white paper asks one organizing question: **given a schema-first, inventory-centric PHP commerce platform designed around canonical JSON inventory records, collectible and one-off goods, portable GI payloads, Crypt viewer presentation, marketplace adapters, and future AI-agent compatibility, which payment architecture best balances cost, simplicity, scalability, user experience, inventory integrity, and future expansion?** The answer reached here is not “pick a gateway first.” It is **canonical data first, payment second**. In practice, the strongest architecture for Physishop is a hybrid: a custom PHP inventory core and canonical schema as the system of record; Stripe Hosted Checkout as the primary payment edge; PayPal as a supplemental wallet and link-based channel; and marketplace adapters that read from the same inventory truth rather than owning it. That conclusion follows from current processor pricing and feature sets, research on checkout cognition and trust, and the growing importance of structured, machine-readable commerce data for search and agentic systems. [1]

The design brief’s recurring themes—canonical data first, schema-driven architecture, inventory → discovery → transaction → feedback loops, cognitive-load reduction, AI-readable structures, modular PHP deployment, multi-channel commerce, Crypt/GI integration, a hybrid record-store/collectibles/art model, and future agent compatibility—fit unusually well with the direction of current web standards and AI tooling. Google’s merchant listing features depend on structured `Product` and `Offer` data, JSON Schema is now being formalized by an active IETF working group after broad industry adoption, OpenAI’s Structured Outputs explicitly target JSON Schema-conformant responses, and MCP has emerged as an open protocol for exposing tools, resources, and prompts to AI systems, including a PHP SDK tier. That combination makes a schema-first inventory core not just a software preference, but a discoverability and automation strategy. [2]

Historical Evolution of Payment Systems

The long history of payment systems suggests that commerce did not begin with modern “payments” at all; it began with **record-keeping, obligation, and trust mediation**. Archaeological literature on Near Eastern clay objects shows that counting and administrative practices predated coinage and even formal writing, while more recent work on the origins of money emphasizes that the old linear story “barter naturally evolved into money” is too simple and remains debated. In other words, payment systems have always been part monetary instrument and part information system. That matters for Physishop because it implies that the deepest layer of a commerce stack is the ledgered description of goods, claims, and state transitions, not merely the checkout button. [3]

Physical and institutional payment rails then layered on top of those accounting functions. Central-bank and Federal Reserve histories show the progression from cash issuance and checks to wire systems, electronic point-of-sale networks, and the Automated Clearing House. Fedwire dates to 1915, while ACH emerged in the 1970s as an electronic substitute for paper checks; the same era also saw the infrastructure that made debit and credit POS transactions scalable. The lesson is structural continuity: every payments innovation preserved the need for clearing, settlement, reconciliation, and dispute handling while changing the speed and interface. [4]

Internet payments added a new layer of abstraction by bundling these rails into developer-facing products such as Checkout pages, payment links, marketplace payouts, and wallet support. Modern gateways now compete less on whether they can authorize a card and more on whether they reduce PCI burden, expose stable APIs, support wallets and subscriptions, and provide reliable event models for fulfillment and reconciliation. That is why the relevant comparison for Physishop is not “Stripe versus PayPal” in the abstract, but “which product best preserves inventory truth while minimizing UI friction and operational fragility.” [5]

Digital asset systems are extending this pattern again. The IMF’s 2025 stablecoin paper reports that stablecoin issuance roughly doubled over the prior two years and has become a major component of the crypto ecosystem, while the ECB moved the digital euro project into its next phase in October 2025. At the same time, the BIS argues that tokenization has transformative potential—especially in architectures centered on tokenized central-bank reserves, commercial bank money, and government bonds—while warning that stablecoins fall short as the mainstay of the monetary system on the tests of singleness, elasticity, and integrity. For Physishop, these developments do not justify building on-chain commerce first; they justify designing **provenance-ready, portable, signed records now**, so that future ownership-transfer models can be added without rewriting the core inventory ontology. [6]

Anatomy of a Payment Transaction

A contemporary card transaction still revolves around a predictable set of actors: customer, merchant, gateway or payment service provider, processor or acquirer, issuing bank, acquiring bank, and card network. U.S. Treasury documentation describing its Card Acquiring Service reduces the lifecycle to two primary stages—authorization, then clearing and settlement—while payment platforms and issuer-side documentation further clarify the intermediate operational steps: authorization places a hold, capture claims the authorized amount, clearing finalizes network records, and settlement moves funds between institutions. Those stages are the backbone around which fraud checks, fulfillment decisions, refunds, and reconciliation workflows are built. [7]

For Physishop, the critical architectural consequence is that the **authorization event is not the order truth**. Customer-facing success pages are useful, but they are not reliable proof of payment completion. Stripe explicitly warns that fulfillment cannot depend solely on the success page because customers may never load it after paying; instead, server-side webhooks are the reliable source of payment events, and Stripe retries deliveries in live mode for up to three days with exponential backoff. That means the canonical order state machine should move from `pending_checkout` to `payment_authorized`, `captured`, `settled`, `refunded`, or `disputed` only from verified server events, not browser confirmation alone. [8]

The data objects named in the brief—customer profile, order record, payment record, fulfillment record, and audit record—therefore map cleanly to best practice. The customer profile should be durable but separable from payment tokens; the order record should mint and retain the canonical inventory item IDs and line-item snapshot at the time of offer or payment; the payment record should preserve gateway IDs, processor status, and settlement fields; the fulfillment record should capture shipping, pickup, or transfer events; and the audit record should preserve all state changes and external callbacks. This is especially important for one-off and collectible inventory, where the system must prove not only that "a payment happened," but exactly **which unique object** moved from available to reserved to sold to delivered to possibly returned. The architecture below expresses that logic in text form.

```
CANONICAL COMMERCE FLOW

Inventory Record
   -> Discovery Surface
      -> Offer / Cart Snapshot
         -> Payment Authorization
            -> Fraud / Risk Review
               -> Capture
                  -> Settlement
                     -> Reconciliation
                        -> Fulfillment
                           -> Feedback / Returns / Provenance Update
```

This flow mirrors official payment-lifecycle descriptions while adding the inventory and provenance layers that a commodity storefront can often ignore but a collectibles platform cannot. [9]

Provider and Channel Landscape

Among mainstream processors, **Stripe** currently offers the clearest fit for a PHP-based, schema-first custom platform. Stripe's standard U.S. domestic card price is 2.9% plus 30¢, Checkout is included in standard pricing, the platform advertises support for 135+ currencies and 100+ payment methods, and Connect is expressly designed for platforms and marketplaces that need to split payments and manage sellers or connected accounts. Stripe Payment Links extend the same rails to low-code or no-code sales surfaces, and Stripe's officially documented checkout flow strongly emphasizes webhooks for reliable fulfillment. In short, Stripe combines low integration friction at the edge with a comparatively strong long-term path for custom orchestration. [10]

PayPal is strong as a supplemental channel because it reduces trust friction for buyers who prefer wallet-based checkout and because its Payment Links and Buttons work well in email, social, QR, and ad-hoc sales contexts. Current U.S. pricing materials show 2.89% + 29¢ for Expanded Checkout card processing, 2.99% + 49¢ for card payments through PayPal Checkout, and 3.49% + 49¢ for PayPal and Venmo wallet transactions. PayPal also supports subscriptions and multi-seller payment patterns through its developer stack. For Physishop, that makes PayPal valuable not as the canonical heart of commerce, but as a high-trust supplemental surface for buyers who would abandon a card form but will complete a wallet flow. [11]

Square is operationally attractive when in-person selling, pop-ups, or POS convergence matter as much as online sales. Square's official API pricing shows 2.9% + 30¢ for Online API payments, its Checkout API creates Square-hosted checkout pages, and its Orders, Payments, and Inventory APIs plus webhooks make it possible to synchronize orders and stock changes. Square is therefore strong if Physishop's future tilts toward fairs, shows, record-store counters, or mixed online/offline events. It is somewhat less compelling if the primary goal is a custom collector-focused web platform with complex canonical metadata and multi-marketplace adapters, because Square's strongest differentiator is its unified seller/POS environment. [12]

Authorize.net remains viable where merchants want a longstanding gateway model, hosted forms, and recurring billing, but it is less elegant for a future-facing inventory core than Stripe. Official pricing lists an all-in-one plan at $25 per month plus 2.9% + 30¢ per transaction. Its Accept Hosted form preserves SAQ-A PCI scope, its ARB tools support subscriptions, and Customer Profiles support tokenized repeat payments and hosted profile editing. Those are real strengths. The tradeoff is that Authorize.net is architecturally solid but less aligned with modern multi-party platform patterns, marketplace economics, and agent-readable product ecosystems than Stripe or Adyen. [13]

Braintree still supports recurring billing and major wallets such as Apple Pay, Google Pay, PayPal, and Venmo, but two current signals make it a weaker choice for a new greenfield Physishop build: its public pricing transparency is poorer than Stripe's or PayPal's, and its own documentation states that the Drop-in SDK moved to unsupported status on July 14, 2026. That does not make Braintree unusable, but it does reduce confidence in it as the cleanest foundation for a modular, long-lived architecture. [14]

Adyen is powerful, especially for enterprises, international payment-method breadth, tokenization, and marketplace fee splitting. Its pricing model is explicitly a fixed processing fee plus payment-method fee, often under interchange++ structures rather than a single flat blended number, and its docs expose mature tokenization and marketplace split mechanics. For Physishop's current brief, however, Adyen's strengths likely exceed the near-term needs unless international complexity or a larger multi-seller marketplace becomes a phase-two requirement. It is a strong scale-up candidate, not the starting point. [15]

The channel layer tells the same story. eBay's Inventory API is explicitly SKU-driven and supports offers, inventory locations, business policies, and even real-time inventory checks. Shopify combines strong checkout, inventory locations, marketplace/social selling, and explicit "agentic commerce" messaging in its current plans, but it also introduces monthly platform costs and, on some plans, third-party transaction fees if Shopify Payments is not used. Etsy's Open API v3 supports inventory, sales orders, shop management, and now even publishes an MCP server for AI-assisted development. WooCommerce exposes products and orders over a JSON REST API and supports multi-currency and a range of payment methods through WooPayments. Amazon's SP-API offers listings, catalog, and orders, while Discogs still matters for record sellers but is operationally more marketplace-specific, with a 9% selling fee and strong CSV/bulk inventory workflows rather than a modern generalized seller platform. For a schema-first Physishop, these channels should be **adapters**, not the system of record. [16]

Cognitive Design and Checkout UX

The checkout section of this paper is not ancillary; it is central. Payment systems do not merely move funds—they allocate user attention under uncertainty. Cognitive research remains consistent on one fundamental constraint: working memory is limited. Nelson Cowan's review places central working-memory capacity for young adults around three to five meaningful items, while later work on chunking shows that

people cope with complexity by recoding smaller units into larger, more familiar ones. Cognitive-load theory similarly argues that system designers should reduce unproductive load so that the user can focus on the task that matters. In checkout terms, this means fewer simultaneous decisions, fewer competing form demands, and a flow that groups related information into intelligible chunks. [17]

Applied to payments, the evidence strongly favors minimizing visible complexity. Baymard's ongoing checkout research reports a current average cart-abandonment rate of 70.19%; its 2024 field-count analysis reports that the average checkout contains 11.3 form fields and that 17% of users have abandoned specifically because checkout felt too long or complicated. Baymard also reports that the average large ecommerce site could increase conversion materially through checkout UX improvements. These statistics are not peer-reviewed journal evidence, but they are high-value industry benchmarks built from large-scale testing and benchmarking. Their operational meaning for Physishop is simple: every avoidable field, forced account wall, or ambiguous payment state is a tax on conversion. [18]

Trust signals matter because online purchases always ask the user to accept some combination of performance risk, privacy risk, payment risk, and merchant uncertainty. Peer-reviewed research in the *International Journal of Information Management*, *Decision Support Systems*, and related journals shows that perceived online-payment and security risks depress purchase intention, and that trust and assurance mechanisms mediate purchase behavior. For Physishop—especially as a seller of art, records, and collectibles where item uniqueness raises stakes—clear return policies, recognizable payment methods, shipping and provenance clarity, and a visibly competent checkout are not cosmetic extras. They are part of the transaction infrastructure. [19]

This is why hosted checkout should be taken seriously rather than dismissed as generic. Stripe Checkout, PayPal's wallet flows, and Square-hosted checkout pages all offload some UI and compliance burden while preserving buyer familiarity. That usually lowers extraneous cognitive load compared to a fully custom card form built too early. The right move is not to design every pixel of payment immediately; it is to reserve custom design energy for the segments where Physishop is actually differentiated: item identity, provenance, reserve logic, scarcity communication, channel syndication, and post-sale ownership history. Let payment UI be boring where boring converts better. [20]

Inventory-Centric Architecture for Physishop and Crypt

The strongest interpretation of the design brief is that **inventory, not checkout, is the primary domain object**. Payment should consume an already-structured offer; it should not be the place where product reality is improvised. JSON Schema's current formalization effort and broad cross-industry use make it an appropriate basis for a canonical inventory model, because it allows constraints, validation, and predictable machine handling across services. In practice, each Physishop item should have a permanent UUID; a canonical JSON record; explicit condition, provenance, price, media, SEO, shipping, and rights fields; one or more channel-mapping fields; and state transitions that are separate from any single storefront or processor. [21]

A text-only architecture sketch makes the dependency order clear:

```
PHYSISHOP CANONICAL STACK
```

```
Canonical Item Record
  - UUID
  - Product / artifact metadata
  - Condition / provenance / edition data
  - Media references
  - SEO + merchant structured data
  - Shipping / tax / reserve fields
  - Channel adapter mappings
  - Audit trail pointers

        ↓

Discovery Surfaces
  - Physishop PHP storefront
  - Search / SEO surfaces
  - eBay / Etsy / Discogs / Amazon / Shopify / Woo adapters
  - GI Card / portable payload views
  - Future agent interfaces

        ↓

Transaction Edge
  - Hosted checkout or payment link
  - Authorization / capture / settlement events
  - Refund / dispute events

        ↓

Inventory State Machine
  available -> reserved -> sold -> fulfilled -> archived / returned

        ↓

Feedback Loop
  analytics -> pricing -> channel tuning -> provenance / ownership updates
```

The discovery layer should then expose this canonical record in formats useful to humans, search engines, marketplaces, and agents. Google's merchant listing and product documentation explicitly reward well-formed `Product` and `Offer` data, including shipping and return information; product variants can be modeled with `ProductGroup`; and organization-level return-policy markup can be surfaced in search. That supports the brief's themes of AI-readable structures, discovery, cognitive load reduction, and future portability. A user should encounter a cleaner page because the underlying data is richer, not because the rich data was omitted. [22]

This is also where the project-specific **Crypt** and **GI card** concepts can be productively interpreted. Public source material does not define those internal terms, so this white paper treats them as supplied design abstractions rather than externally standardized systems. Within that frame, the best public analogue is a

portable, signed, schema-validated payload that can be rendered in multiple UIs: storefront page, marketplace export, QR or record card, post-sale certificate, or future machine-to-machine exchange. That interpretation is strengthened by current AI tooling: OpenAI's Structured Outputs are built around schemas, the Agents SDK is moving toward safe long-horizon tool use in controlled workspaces, MCP standardizes tool and resource access for LLM applications, and Etsy's own developer program now advertises an MCP server. The implication is an inference, but a strong one: **canonical structured inventory is the shortest path to agent-compatible commerce**. [23]

Recommended Architecture for Physishop

The evidence supports the following evaluation. **Option A, simple PayPal buttons**, is useful for rapid deployment and high-trust lightweight sales surfaces, especially in social, email, QR, or low-code contexts, but it is too thin to function as the primary long-term architecture for a collectible inventory system whose key asset is canonical item state. **Option B, Stripe Hosted Checkout**, is the strongest low-friction baseline because it gives a mature hosted payment UI at standard Stripe rates with no added checkout fee and a strong webhook model. **Option C, Stripe plus a custom inventory layer**, is better than Option B whenever the merchant cares about unique goods, inventory reservation, cross-channel orchestration, or future marketplace and AI adapters. **Option D, Shopify backend**, is excellent if the business chooses operational convenience, multichannel retail tooling, and Shopify's broader commerce ecosystem over domain-level control, but it raises platform dependency, recurring cost, and some payment-governance constraints. **Option E, custom PHP plus Stripe plus Crypt**, offers the most strategic control, but imposes more engineering burden than many early phases need. **Option F, the hybrid Physishop architecture**, which uses a custom PHP canonical inventory core and Crypt/GI presentation layer while delegating the payment edge to Stripe Hosted Checkout and selectively to Payment Links or PayPal, is the best balance overall. [24]

The recommended implementation is therefore:

1. **Canonical item schema in PHP first.** Every saleable object gets a UUID-based JSON record that drives storefront rendering, SEO markup, channel adapters, reserve logic, and auditability.
2. **Stripe Hosted Checkout as default checkout surface.** This reduces PCI and UI burden while preserving conversion and providing reliable server events for fulfillment.
3. **Stripe Payment Links for lightweight and off-site sales.** Use these for social, direct-message, event, QR, or ad-hoc sales where a full product page is unnecessary.
4. **PayPal as a supplemental rail, not the source of truth.** Offer it where wallet trust matters, but let the canonical inventory/payment state live in Physishop.
5. **Marketplace adapters as readers and writers against canonical inventory.** eBay, Etsy, Discogs, Shopify, WooCommerce, and Amazon integrations should map to internal item truth rather than maintain parallel conflicting item records.
6. **Reservation and reconciliation as first-class logic.** One-off and scarce goods should not enter "sold" state at cart creation; they should move through explicit reserve and payment-confirmed transitions.
7. **Crypt/GI outputs as portable projections of canonical records.** Certificates, viewer pages, QR outputs, and ownership-transfer models should all be generated from the same payload family. [25]

In plain language, the answer to the paper's central question is this: **the optimal payment processing architecture for Physishop is not a monolithic commerce platform and not a bare button strategy. It is a hybrid architecture centered on a schema-driven PHP inventory core with Stripe Hosted**

Checkout as the primary payment interface, Stripe Payment Links for lightweight channel sales, and PayPal as an optional wallet overlay. That combination preserves cost discipline, keeps the UI conversion-strong, protects inventory integrity for one-off goods, and leaves room for channel syndication, provenance-rich collectible sales, and future agent automation without ceding the core data model to any single marketplace or processor. [26]

Open questions and limitations. Because this draft intentionally did not pull directly from the project folders, terms such as **Crypt**, **GI card**, and any internal payload conventions are interpreted from the brief rather than documented from source files. That means the public research can justify the architecture pattern—canonical schema, hosted payment edge, portable structured payloads, provenance-ready records—but cannot verify internal field names, signing methods, or existing adapter code. If those internal specifications materially differ from the assumptions used here, the recommendation would still likely remain "schema-first hybrid with Stripe primary," but the exact payload and event design should be conformed to the project's actual canonical schema. [27]

Works Cited

Adrian, Tobias, et al. *Understanding Stablecoins*. International Monetary Fund, 2025.

Baymard Institute. "Cart & Checkout Usability Research." Baymard Institute, 2026.

Baymard Institute. "Checkout Optimization: 5 Ways to Minimize Form Fields in Checkout." Baymard Institute, 2024.

Bennison-Chapman, Lucy E. "Reconsidering 'Tokens': The Neolithic Origins of Accounting or Multifunctional, Utilitarian Tools?" *Cambridge Archaeological Journal*, vol. 29, no. 2, 2019, pp. 233–259.

Bureau of the Fiscal Service. "How CAS Works." U.S. Department of the Treasury, 2026.

Cowan, Nelson. "The Magical Mystery Four: How Is Working Memory Capacity Limited, and Why?" *Current Directions in Psychological Science*, vol. 19, no. 1, 2010, pp. 51–57.

European Central Bank. "Digital Euro." ECB, updated 2026.

Fauvelle, Mikael. "The Trade Theory of Money: External Exchange and the Origins of Money." *Journal of Archaeological Method and Theory*, vol. 32, 2025.

Federal Reserve History. "Payment Systems." Federal Reserve Bank of St. Louis, 2026.

Federal Reserve History. "Automated Clearing House Payments." Federal Reserve Bank of St. Louis, 2023.

Google Search Central. "Merchant Listing (`Product`, `Offer`) Structured Data." Google for Developers, 2025.

Google Search Central. "Introduction to `Product` Structured Data." Google for Developers, 2025.

Google Search Central. “Merchant Return Policy Structured Data.” Google for Developers, 2025.

Hong, Ilyoo B., and Hoon S. Cha. “The Mediating Role of Consumer Trust in an Online Merchant in Predicting Purchase Intention.” *International Journal of Information Management*, vol. 33, no. 6, 2013, pp. 927–939.

Hsu, Li-Ling, et al. “Determinants of Successful Online Transactions: Effects of Transaction Assurance Seal and Reputation Rating Affecting Trust and Purchase Intention of Consumers.” *Human Systems Management*, vol. 34, no. 2, 2015.

IETF JSON Schema Working Group. “JSON Schema Working Group Charter.” Internet Engineering Task Force, 2026.

Model Context Protocol. “SDKs.” Model Context Protocol, 2026.

OpenAI. “Introducing Structured Outputs in the API.” OpenAI, 2024.

OpenAI. “Structured Model Outputs.” OpenAI API Documentation, 2026.

OpenAI. “The Next Evolution of the Agents SDK.” OpenAI, 2026.

Paas, Fred, and Jeroen J. G. van Merriënboer. “Cognitive-Load Theory: Methods to Manage Working Memory Load in the Learning of Complex Tasks.” *Current Directions in Psychological Science*, vol. 29, no. 4, 2020, pp. 394–398.

Schema.org. “Offer.” Schema.org, 2026.

Square. “Payments Pricing with Square APIs and SDKs.” Square Developer Documentation, 2026.

Square. “Checkout API.” Square Developer Documentation, 2026.

Stripe. “Pricing & Fees.” Stripe, 2026.

Stripe. “Stripe Checkout.” Stripe, 2026.

Stripe. “Platforms and Marketplaces with Stripe Connect.” Stripe Documentation, 2026.

Thalmann, Mirko, Alessandra S. Souza, and Klaus Oberauer. “How Does Chunking Help Working Memory?” *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 45, no. 1, 2019, pp. 37–55.

WooCommerce. “WooCommerce REST API Documentation.” WooCommerce, 2026.

World Bank. *The Global Findex Database 2025*. World Bank, 2025.

1 10 24 26 Pricing & Fees
https://stripe.com/us/pricing?utm_source=chatgpt.com

2 22 27 How To Add Merchant Listing Structured Data | Google Search Central | Documentation | Google for Developers
https://developers.google.com/search/docs/appearance/structured-data/merchant-listing?hl=en&utm_source=chatgpt.com

3 Reconsidering 'Tokens': The Neolithic Origins of Accounting or Multifunctional, Utilitarian Tools? | Cambridge Archaeological Journal | Cambridge Core
https://www.cambridge.org/core/journals/cambridge-archaeological-journal/article/reconsidering-tokens-the-neolithic-origins-of-accounting-or-multifunctional-utilitarian-tools/7E6C04CB040AD8AA0EA84B94D4D275C4?utm_source=chatgpt.com

4 Payment Systems | Federal Reserve History
https://www.federalreservehistory.org/topics/payment-systems?utm_source=chatgpt.com

5 20 25 Stripe Checkout | Checkout Pages for Your Website
https://stripe.com/payments/checkout?utm_source=chatgpt.com

6 Understanding Stablecoins
https://www.imf.org/en/publications/departmental-papers/issues/2025/12/02/understanding-stablecoins-570602?utm_source=chatgpt.com

7 9 How CAS Works | Bureau of the Fiscal Service
https://fiscal.treasury.gov/payments-to-government/card-acquiring-service-cas/how-cas-works?utm_source=chatgpt.com

8 Customize redirect behavior | Stripe Documentation
https://docs.stripe.com/payments/checkout/custom-success-page?utm_source=chatgpt.com

11 PayPal Business Pricing | Transaction & Processing Fees | PayPal US
https://www.paypal.com/us/business/fees?utm_source=chatgpt.com

12 Payments Pricing with Square APIs and SDKs
https://developer.squareup.com/docs/payments-pricing?utm_source=chatgpt.com

13 Plans and pricing | Authorize.net
https://www.authorize.net/sign-up/pricing.html?utm_source=chatgpt.com

14 Braintree SDK Docs
https://developer.paypal.com/braintree/articles/guides/recurring-billing/overview?utm_source=chatgpt.com

15 Pricing for supported payment methods - Adyen
https://www.adyen.com/pricing?pix=6f_0_0&utm_source=chatgpt.com

16 Inventory API Overview | eBay Developers Program
https://developer.ebay.com/api-docs/sell/inventory/overview.html?utm_source=chatgpt.com

17 The Magical Mystery Four: How is Working Memory Capacity Limited, and Why? - PMC
https://pmc.ncbi.nlm.nih.gov/articles/PMC2864034/?utm_source=chatgpt.com

18 E-Commerce Cart & Checkout Usability Research – Baymard
https://baymard.com/research/checkout-usability?utm_source=chatgpt.com

19 The mediating role of consumer trust in an online merchant in predicting purchase intention - ScienceDirect
https://www.sciencedirect.com/science/article/pii/S0268401213000984?utm_source=chatgpt.com

21 JSON Schema (jsonschema)
https://datatracker.ietf.org/wg/jsonschema/about/?utm_source=chatgpt.com

23 The next evolution of the Agents SDK | OpenAI
https://openai.com/index/the-next-evolution-of-the-agents-sdk/?utm_source=chatgpt.com