Advanced Architectures for High-Resiliency E-Commerce Data Extraction: Overcoming PerimeterX Human Bot Defender on Walmart Canada Infrastructure

The contemporary digital landscape for automated data collection has evolved into a sophisticated theater of technical conflict, where e-commerce giants like Walmart Canada employ advanced behavioral biometrics to safeguard proprietary pricing and inventory data. At the center of this defensive perimeter is the PerimeterX system, recently rebranded as HUMAN, which represents a paradigm shift from traditional firewall-based blocking to dynamic, score-based trust evaluation.[1] For engineers developing grocery store web scrapers, the emergence of the "Press and Hold" challenge signifies a failure in the scraper's ability to maintain a human-identical digital signature across the network, transport, and application layers.[1] Achieving a successful and sustainable bypass requires more than simple proxy rotation; it necessitates the orchestration of high-fidelity browser environments, cryptographic impersonation, and the simulation of natural human physiological interactions.[3]

The Probabilistic Engine of PerimeterX Bot Defender

PerimeterX does not operate through static blacklists or simple signature matching but instead functions as a continuous monitoring suite that assigns a trust score to every client interaction.[1] This trust score is updated in real-time as the client navigates the site, and the "Press and Hold" CAPTCHA is triggered specifically when behavioral or environmental signals fall into a range that suggests automation.[2] Most scrapers encounter blocks because they focus exclusively on delivering a valid HTTP response while ignoring the invisible telemetry challenges that PerimeterX executes in the background.[1] These challenges measure how the browser environment processes content, reports hardware capabilities, and reacts to timing intervals.[1]

The system tracks timing patterns, JavaScript execution signals, header consistency, and session tokens to build a comprehensive profile of what constitutes a real user.[1] Standard scraping libraries like Python's requests or Node.js's native http module are often flagged immediately because they lack a JavaScript execution environment, failing the initial sensor challenges that PerimeterX injects into the page.[1] Even when using browser automation tools like Playwright or Selenium, the default configurations of these tools leave behind "digital

fingerprints" that are easily identified by anti-bot algorithms.[3]

Detection Mechanism	Data Points Collected	Impact on Scraper Integrity
IP Reputation	ASN type, history of malicious activity, proxy type (Data Center vs. Residential).[1]	High; initial filter that determines the likelihood of further scrutiny.
Browser Fingerprinting	Canvas rendering, WebGL vendor, font lists, screen resolution, audio context.[3]	High; identifies the use of headless browsers or inconsistent hardware reporting.
TLS/JA3 Handshake	Cipher suites, extension order, protocol versions, elliptic curves.[8]	Medium; identifies non-standard HTTP clients before the application layer is reached.
Behavioral Biometrics	Mouse velocity, acceleration, typing cadence, scroll patterns.[2]	Very High; primary trigger for "Press and Hold" challenges after session initialization.

The holistic nature of this detection strategy means that a single inconsistency, such as a high-performance residential IP address paired with a generic "HeadlessChrome" browser fingerprint, creates a mismatch that lowers the trust score.[1] Successful scraping of Walmart Canada requires ensuring that every layer of the client identity is internally consistent and externally plausible for a Canadian retail consumer.[11]

Residential Proxy Infrastructure and Regional Authenticity

IP reputation is the foundational layer of the PerimeterX defense stack. For a domestic retailer like Walmart Canada, the system expects traffic to originate from residential Internet Service Providers (ISPs) located within Canada.[11] Data center IPs, which are associated with cloud infrastructure like AWS or DigitalOcean, are treated with extreme suspicion because they are rarely used by real consumers for grocery shopping.[1] Residential proxies, ethically sourced from actual home addresses, provide the necessary "camouflage" to blend in with legitimate

user traffic.[11]

Strategic Utilization of Mobile and Residential IPs

Mobile proxies represent the highest tier of IP authenticity. Because mobile network operators (MNOs) use Carrier-Grade NAT (CGNAT) to share a single IP address among thousands of mobile devices, PerimeterX is often restricted from blocking these IPs entirely, as doing so would negatively impact thousands of legitimate customers.[2] For highly sensitive operations or the initial authentication phase of a scraper, mobile proxies provide a nearly unblockable origin.[2]

Residential proxies are more cost-effective for high-volume data extraction but require sophisticated rotation logic. It is insufficient to simply rotate the IP; the scraper must ensure that the IP location remains consistent with the regional headers and cookies sent to the Walmart API.[13] For example, a request targeting a store in Calgary, Alberta, using the postal code T2P 2M5, should ideally originate from an IP address within the Alberta region to avoid cross-regional inconsistencies.[13]

Proxy Provider	Global Pool Size	Protocol Support	Primary Use Case for Walmart Canada
Oxylabs	175M+ IPs [15]	HTTP, HTTPS, SOCKS5 [16]	Enterprise-scale scraping with precise city and ASN targeting.[12]
Bright Data	72M+ IPs [17]	HTTP, HTTPS, SOCKS5 [17]	High-uptime requirements (~99.9%) and robust proxy management tools.[12]
Decodo (Smartproxy)	115M+ IPs [16]	HTTP, HTTPS, SOCKS5 [16]	Mid-tier projects requiring high speed and competitive per-GB pricing.[12]
NetNut	85M+ IPs [11]	HTTP, HTTPS [11]	Large-scale data

			collection using direct ISP-connectivity for low latency.[11]

When implementing residential proxies, the use of "backconnect" proxies is standard. These provide a single entry point (gateway) that automatically handles the rotation of outbound IPs.[16] However, for scraping tasks that require multiple steps—such as searching for a product and then visiting individual product pages—it is critical to use "sticky sessions" to maintain the same IP for the duration of the task, preventing PerimeterX from flagging the session for unnatural IP switching.[12]

Cryptographic Impersonation via TLS and HTTP/2 Fingerprinting

A significant vulnerability in modern scrapers is the transport layer. Before any application data is exchanged, the TLS handshake reveals the "cryptographic identity" of the client.[9] PerimeterX uses the JA3 algorithm to analyze the ClientHello packet, creating a unique MD5 hash based on the TLS version, accepted cipher suites, and extensions.[9] Standard Node.js or Python libraries have static JA3 fingerprints that differ drastically from those of a real browser, allowing PerimeterX to block the connection before the HTML is even requested.[18]

JA3 and JA4 Fingerprint Synthesis

The JA3 fingerprint is composed of five parameters: TLSVersion, Ciphers, Extensions, SupportedGroups, EllipticCurvePointFormats.[9] Real browsers frequently update these parameters. For instance, Chrome might prioritize specific encryption algorithms like ChaCha20 on mobile but prefer AES-GCM on desktop.[9] Furthermore, major browsers have begun randomizing the order of extensions (known as GREASE values) to prevent stable fingerprinting, which has led to the development of the JA4 algorithm.[9] JA4 addresses this by sorting extensions alphabetically before hashing, creating a more stable identifier for specific browser families.[9]

To bypass this, scraper developers must use specialized libraries that can impersonate browser-identical TLS stacks. In Python, libraries such as curl_cffi or tls-client allow the scraper to specify a browser profile (e.g., "chrome_120") and perfectly mimic the handshake.[18] In Node.js, similar results can be achieved by using cycle-tls or by routing traffic through a TLS-aware proxy that terminates the TLS connection and re-establishes it with a browser-like signature.[4] Failure to match the TLS fingerprint leads to an immediate 403 Forbidden error, which is often misinterpreted by developers as an IP block.[2]

HTTP/2 Frame Fingerprinting

Beyond the TLS layer, the HTTP/2 protocol itself is a vector for detection. When establishing an HTTP/2 connection, the client sends a "SETTINGS" frame that includes parameters like SETTINGS_HEADER_TABLE_SIZE and SETTINGS_MAX_CONCURRENT_STREAMS.[8] The specific values and the order in which these settings are sent are highly characteristic of the browser engine.[8] PerimeterX monitors these HTTP/2 signatures to identify discrepancies, such as a client claiming to be "Chrome" but sending settings typical of a Go-based HTTP library.[8] Ensuring high-resiliency requires a scraper that supports HTTP/2 and mirrors the settings of the targeted browser environment.[2]

Advanced Browser Fingerprinting and Stealth Tactics

Even with a perfect residential IP and a valid TLS handshake, PerimeterX can still identify a scraper through application-layer fingerprinting.[1] When a browser loads a page, PerimeterX executes JavaScript to probe the environment for automation markers.[3] This process creates a "digital signature" that identifies the device hardware and software configuration.[3]

Navigating the Sensor Challenges

PerimeterX's sensor challenges are designed to detect inconsistencies. For example, the navigator.webdriver property is a standard indicator that a browser is being controlled by an automation framework.[3] While stealth plugins can hide this property, advanced detectors look for "side-channel" leaks. A common leak is the inconsistency between the User-Agent and the WebGL renderer.[3] If a User-Agent reports a high-end Windows machine, but the WebGL renderer identifies a generic "Mesa" driver often used in Linux-based headless environments, the scraper is flagged.[3]

Fingerprinting Category	Specific Probes	Mitigation Strategy
Automation Indicators	navigator.webdriver, window.__nightmare, global variables from Selenium/Puppeteer.[3]	Use playwright-stealth or undetected-chromedriver to strip these markers.[20]
Rendering Engines	Canvas 2D image hashing, WebGL unmaskedRenderer and unmaskedVendor.[3]	Inject realistic WebGL signatures (e.g., Apple GPU or Intel Iris) using fingerprint-injector.[7]

Hardware Environment	hardwareConcurrency, screen resolution, color depth, audio latency.[3]	Override these properties to match the target device profile consistently.[6]
Timing & Runtime	JavaScript engine quirks, execution speed, font rendering timing.[1]	Use real browser instances instead of simulated environments; avoid heavy CPU throttling.[2]

Implementing Fortified Browsers

To achieve "sophisticated stealth," developers should utilize fortified versions of headless browsers. The playwright-with-fingerprints plugin is an advanced tool that goes beyond simple property masking.[6] It inherits the Playwright API but replaces suspicious browser properties with values from a database of real-world browser fingerprints.[6] This ensures that all components of the fingerprint—User-Agent, WebGL, screen resolution, and available fonts—are internally consistent and derived from a genuine hardware configuration.[6]

Behavioral Biometrics and the "Press and Hold" Mechanism

The most challenging component of PerimeterX is the behavioral analysis module. Unlike static CAPTCHAs, "Press and Hold" is a behavioral challenge that tests for the presence of a human motor system.[1] Automated clicks are often instantaneous and occur exactly at the center of an element, which is a "dead giveaway" to anti-bot systems.[10]

The Physics of Human-Like Interaction

Human behavior is characterized by "chaos" and non-linear patterns. A human user does not navigate from the search bar to a product with millisecond precision.[2] To mimic this, a scraper must implement:

1. **Non-Linear Mouse Movements**: Standard page.click() methods in Playwright jump the cursor instantly to the target.[10] Tools like ghost-cursor generate a series of points along a smooth curve to simulate human hand motion, including physics-based acceleration, overshooting the target, and subsequent micro-corrections.[10]
2. **Physiological Timing**: Automated scripts often perform actions with a fixed periodicity. A human user, however, exhibits variable delays.[2] Implementing a randomized "typing cadence" with page.keyboard.type() (e.g., 50ms to 200ms between keys) and adding random "contemplation" pauses between page actions makes the behavior appear much more human.[23]
3. **Browsing Entropy**: Robots typically follow a direct path to the data. Humans browse in more chaotic patterns, often scrolling through irrelevant sections, hovering over images,

or clicking on breadcrumbs.[2] Incorporating these "filler" actions increases the trust score of the session.[2]

Programmatic Solving vs. Avoidance

While it is possible to programmatically solve the "Press and Hold" button by locating the element and simulating a long press (page.mouse.down(), waitForTimeout(3000), page.mouse.up()), PerimeterX often detects the lack of micro-tremors or natural jitter during the hold.[28] Therefore, the most effective strategy is avoidance through behavioral stealth.[1] If the trust score remains high due to excellent IP and fingerprint management, the challenge may never be presented.[1]

If a challenge is unavoidable, integrating an AI-powered CAPTCHA solving service like CapSolver or 2Captcha is necessary.[23] These services use machine learning to solve the challenge and return a token that can be injected into the page context, allowing the scraper to proceed.[31]

Reverse Engineering Walmart Canada's Internal Data Architecture

Walmart Canada utilizes a complex web and mobile architecture that offers multiple avenues for data extraction. Traditional scrapers often rely on parsing the HTML of search results pages, but this is the most resource-intensive and highly-monitored layer of the site.[33]

The GraphQL Orchestra Endpoint

The Walmart Canada mobile app and parts of the modern web interface utilize a GraphQL architecture to fetch data.[13] Intercepting mobile traffic using tools like Charles Proxy reveals an endpoint located at walmart.ca/orchestra/snb/graphql/search.[13] This "Orchestra" layer is an internal API that allows for structured queries, returning clean JSON data without the overhead of HTML rendering.[13]

The GraphQL query for product searches, often named getPreso, accepts several variables that control the search scope:

- qy: The search term.[35]
- cat_id: The category identifier.[35]
- prg: The platform identifier, frequently set to ios or android to trigger mobile-specific responses.[35]
- p13n: Personalization data, including the device type and a unique visitor ID (vid).[35]

By targeting these GraphQL endpoints directly, a scraper can retrieve product names, unit prices (e.g., $1.37/100g), and stock levels with significantly higher efficiency.[35] However, these

API calls must be accompanied by the correct authorization headers, such as TTD-Auth, and regional cookies like deliveryCatchment to ensure the data is accurate for the desired Canadian locale.[13]

Extracting Hidden Web Data

For the desktop web interface, Walmart embeds the initial state of the page within the HTML in a JSON format known as "Hidden Web Data".[33] This is typically found within a script tag with the ID __NEXT_DATA__.[33] Instead of writing complex XPath or CSS selectors to parse individual product elements, the scraper can extract the entire JSON object from this tag.[33] This object contains the complete product list, pricing, and variant information (size, color, pack size) in a structured format that is much less prone to breakage from UI changes.[33]

Rate Limiting, Lifecycle Management, and Operational Scaling

A scraper that works for ten requests might fail at ten thousand due to rate limiting and session fatigue.[25] Walmart and PerimeterX monitor the volume of traffic from specific IP ranges and session tokens, implementing throttling when thresholds are exceeded.[2]

Strategic Rate Limiting and Backoff

Successful scaling requires the implementation of exponential backoff logic.[33] When the scraper receives a 403 Forbidden or 429 Too Many Requests response, it should not immediately retry, as doing so confirms the automated nature of the traffic.[33] Instead, the delay between retries should increase (e.g., 2s, 4s, 8s, 16s) while simultaneously rotating the proxy and user-agent.[33]

Scaling Challenge	Mitigation Strategy
IP Throttling	Use large residential proxy pools and rotate the IP after a fixed number of requests or a session timeout.[2]
Session Expiry	Monitor for 401 Unauthorized or redirect-to-login responses; refresh authentication tokens and clear cookies periodically.[1]
Regional Logic	Store-specific data is tied to cookies like

	walmart.nearestPostalCode; ensure these are set correctly for each regional search (e.g., Calgary vs. Toronto).[13]
Parallelization	Use PlaywrightCrawler or similar frameworks to run multiple browser contexts in parallel, each with its own proxy and fingerprint.[13]

Session Lifecycle and Cookie Management

PerimeterX uses cookies and session tracking to link multiple requests into a single behavioral narrative.[25] If a single session remains active for hours and visits thousands of pages in a perfectly linear fashion, it will be flagged.[1] A high-resiliency strategy involves "session cycling," where the scraper completely clears its cookies, local storage, and rotates its fingerprint every 50 to 100 requests, effectively "rebirthing" as a new user.[1]

For regional grocery data, it is critical to manage the following cookies carefully:

- walmart.nearestPostalCode: This ensures the API returns the correct inventory for the user's specific store.[13]
- defaultNearestStoreId: Directly specifies the store location.[13]
- deliveryCatchment: Affects the estimated delivery dates and fulfillment options displayed in the data.[13]

Conclusion and Recommended Implementation Framework

To ensure that the Walmart Canada scraper operates successfully and reliably against PerimeterX's "Press and Hold" protections, a unified architectural approach is required. The solution is not found in a single tool but in the integration of five distinct technical domains: network authenticity, cryptographic identity, application-layer stealth, behavioral biometrics, and intelligent data targeting.

The primary recommended framework for this scraper is built on Playwright (Node.js) using the following configuration:

1. **Identity Layer**: Integrate premium Canadian residential proxies (e.g., Oxylabs or Bright Data) with city-level targeting and sticky session support to maintain regional pricing consistency.[12]
2. **Stealth Layer**: Utilize the playwright-with-fingerprints plugin to automate the injection of high-entropy, consistent browser fingerprints, ensuring that WebGL, Canvas, and Hardware markers match real-world devices.[3]

3. **Cryptographic Layer**: Route traffic through a TLS-aware proxy or use a TLS-emulating library to ensure JA3 and HTTP/2 fingerprints perfectly mimic a standard Chrome on Windows environment.[9]
4. **Behavioral Layer**: Implement ghost-cursor for all click and navigation events and utilize randomized typing cadence for any input fields to evade behavioral biometric triggers.[10]
5. **Data Layer**: Bypass traditional HTML scraping in favor of extracting the __NEXT_DATA__ JSON object from web pages or targeting the internal GraphQL orchestra endpoints for cleaner and more efficient data retrieval.[13]

By synthesizing these strategies into a single narrative of human-like interaction, the scraper will achieve a trust score that bypasses the "Press and Hold" mechanism entirely. For organizations requiring extreme scale with minimal maintenance, leveraging managed scraping APIs like ZenRows or ScrapFly provides a robust alternative by offloading the entire anti-bot bypass logic to specialized third-party infrastructure.[1] Regardless of the chosen path, the key to success lies in maintaining a digital presence that is indistinguishable from a genuine Canadian consumer browsing for their weekly groceries.

Works cited

1. How to Bypass PerimeterX (HUMAN) Bot Detection with ScraperAPI, accessed on January 12, 2026, https://www.scraperapi.com/blog/scrape-perimeterx-protected-websites-with-python/
2. How to Bypass PerimeterX when Web Scraping in 2026 - Scrapfly, accessed on January 12, 2026, https://scrapfly.io/blog/posts/how-to-bypass-perimeterx-human-anti-scraping
3. How to Bypass PerimeterX in 2026: The 4 Best Methods - ZenRows, accessed on January 12, 2026, https://www.zenrows.com/blog/perimeterx-bypass
4. TLS Fingerprinting: How It Works & How to Bypass It (2025) - Browserless, accessed on January 12, 2026, https://www.browserless.io/blog/tls-fingerprinting-explanation-detection-and-bypassing-it-in-playwright-and-puppeteer
5. Walmart press and hold captcha/bot bypass : r/webscraping - Reddit, accessed on January 12, 2026, https://www.reddit.com/r/webscraping/comments/1nguk57/walmart_press_and_hold_captchabot_bypass/
6. Playwright Fingerprinting: Explained & Bypass - ZenRows, accessed on January 12, 2026, https://www.zenrows.com/blog/playwright-fingerprint
7. How to Bypass Browser Fingerprinting With fingerprint-suite - ZenRows, accessed on January 12, 2026, https://www.zenrows.com/blog/fingerprint-suite
8. tls-fingerprinting · GitHub Topics, accessed on January 12, 2026, https://github.com/topics/tls-fingerprinting
9. What is TLS Fingerprint and How to Bypass it in 2025 - Roundproxies, accessed on January 12, 2026, https://roundproxies.com/blog/what-is-tls-fingerprint/

10. How to Use Ghost Cursor for Web Scraping - ZenRows, accessed on January 12, 2026, https://www.zenrows.com/blog/ghost-cursor
11. Best Walmart Proxies in 2025 - ProxyEmpire, accessed on January 12, 2026, https://proxyempire.io/best-walmart-proxies/
12. 7 Best Canada Residential Proxy Providers [2025 Update] - NodeMaven, accessed on January 12, 2026, https://nodemaven.com/blog/canada-residential-proxy/
13. grocery-app/HACKING.md at main · snacsnoc/grocery-app · GitHub, accessed on January 12, 2026, https://github.com/snacsnoc/grocery-app/blob/main/HACKING.md
14. Walmart Canada | BBB Business Profile | Better Business Bureau, accessed on January 12, 2026, https://www.bbb.org/ca/ab/calgary/profile/department-stores/walmart-canada-0107-1072586/addressId/106464
15. Best Residential Proxies in Canada of 2026 - Reviews & Comparison - SourceForge, accessed on January 12, 2026, https://sourceforge.net/software/residential-proxies/canada/
16. The Best Residential Proxies of 2026: Tested & Ranked - Proxyway, accessed on January 12, 2026, https://proxyway.com/best/residential-proxies
17. 9 Best Residential Proxies in 2026 - ZenRows, accessed on January 12, 2026, https://www.zenrows.com/blog/best-residential-proxy
18. Overcoming TLS Fingerprinting in Web Scraping - Rayobyte, accessed on January 12, 2026, https://rayobyte.com/blog/tls-fingerprinting/
19. Bypass Cloudflare TLS Fingerprinting: Advanced Techniques | Kite Metric, accessed on January 12, 2026, https://kitemetric.com/blogs/conquering-cloudflare-s-tls-fingerprinting-advanced-techniques-and-solutions
20. Bypass Proxy Detection with Browser Fingerprint Impersonation - Scrapfly, accessed on January 12, 2026, https://scrapfly.io/blog/posts/bypass-proxy-detection-with-browser-fingerprint-impersonation
21. How to Bypass CAPTCHA With Playwright 2025 - Oxylabs, accessed on January 12, 2026, https://oxylabs.io/blog/playwright-bypass-captcha
22. lisazhuji/shop/How To Bypass Anti-Bots in 2025.md at main - GitHub, accessed on January 12, 2026, https://github.com/berchtsconyz/lisazhuji/blob/main/shop/How%20To%20Bypass%20Anti-Bots%20in%202025.md
23. Bypassing CAPTCHA with Playwright | ScrapingAnt, accessed on January 12, 2026, https://scrapingant.com/blog/bypass-captcha-playwright
24. Make Scraping Human: Ghost Cursor & Puppeteer Guide (2025) - Scrapeless, accessed on January 12, 2026, https://www.scrapeless.com/en/blog/ghost-cursor
25. How to Bypass PerimeterX with Playwright - ScrapeOps, accessed on January 12, 2026, https://scrapeops.io/playwright-web-scraping-playbook/nodejs-playwright-bypass-perimeterx/
26. Xetera/ghost-cursor: 🖱️ Generate human-like mouse ... - GitHub, accessed on

January 12, 2026, https://github.com/Xetera/ghost-cursor
27. How to Bypass CAPTCHA with Playwright: An In-Depth Guide | by James Smith | Medium, accessed on January 12, 2026, https://medium.com/@w908683127/how-to-bypass-captcha-with-playwright-an-in-depth-guide-71b0b08e61b5
28. TIGER-SMS/tab/How to Bypass the “Please Verify You Are a Human” Challenge.md at main - GitHub, accessed on January 12, 2026, https://github.com/hm266386/TIGER-SMS/blob/main/tab/How%20to%20Bypass%20the%20%E2%80%9CPlease%20Verify%20You%20Are%20a%20Human%E2%80%9D%20Challenge.md
29. how to simulate long press using playwright [closed] - Stack Overflow, accessed on January 12, 2026, https://stackoverflow.com/questions/79382807/how-to-simulate-long-press-using-playwright
30. How to long press (Press and Hold) mouse left key using only Selenium in Python, accessed on January 12, 2026, https://stackoverflow.com/questions/68636955/how-to-long-press-press-and-hold-mouse-left-key-using-only-selenium-in-python
31. How to Bypass reCAPTCHA and Turnstile in Crawlee with CapSolver - DEV Community, accessed on January 12, 2026, https://dev.to/luisgustvo/how-to-bypass-recaptcha-and-turnstile-in-crawlee-with-capsolver-307p
32. How to Solve CAPTCHA in Browser-use with CapSolver API, accessed on January 12, 2026, https://www.capsolver.com/blog/All/browser-use-capsolver
33. How to Scrape Walmart.com Product Data (2026 Update) - Scrapfly, accessed on January 12, 2026, https://scrapfly.io/blog/posts/how-to-scrape-walmartcom
34. GraphQL API Queries - TTD Partner Portal, accessed on January 12, 2026, https://partner.dsp.walmart.com/v3/portal/resources/doc/GqlApiQueries
35. Search multiple grocery stores at once | Easton's stuff, accessed on January 12, 2026, https://geekness.eu/node/452
36. GraphQL API Calls - TTD Partner Portal - Walmart, accessed on January 12, 2026, https://partner.dsp.walmart.com/v3/portal/api/doc/GqlApiCallsPlatform
37. Walmart Product Search API - Unwrangle Documentation, accessed on January 12, 2026, https://docs.unwrangle.com/walmart-product-search-api/
38. Walmart Supercenter in Calgary, AB | Grocery, Electronics, Toys, Clothing, Furniture, Baby | Store 3151, accessed on January 12, 2026, https://www.walmart.ca/en/store/3151