ETECHFLOW • MAGENTO 2 / ADOBE COMMERCE

Cache Warmer

Full Page Cache warming · Smart priority queue · Auto-warm on flush · Built-in FPC test

Keeps your storefront fast by pre-loading the Full Page Cache before shoppers arrive. It crawls a smart, prioritised queue of your most valuable URLs, warms them on a schedule, and automatically re-warms any page the moment its cache is cleared — so visitors land on a warm HIT instead of a slow cold render.

ETechFlow_CacheWarmer · Module Documentation & User Guide

Compatible with Magento / Adobe Commerce 2.4.6 – 2.4.9 · PHP 8.1 – 8.4 · Built-in FPC & Varnish · Part of the Etechflow Suite

Table of Contents

1 · Overview

1.1 What the module does

Magento's Full Page Cache (FPC) makes pages fast — but only **after** a page has been visited once and cached. The very first visitor to a cold page (after a deploy, a cache flush, or a content edit) waits for a slow full render. **Etechflow Cache Warmer** removes that cold-start penalty: it builds a prioritised queue of your important URLs, requests each one so Magento caches it, and keeps the cache warm automatically — on a schedule and the instant any page is invalidated. Shoppers consistently land on a warm cache **HIT**.

1.2 Highlights

- **Smart priority queue** — gathers URLs from the sitemap, products, categories and CMS pages, scores each by page type and best-seller popularity, and warms the highest-value pages first.
- **Warming rules** — boost the priority of any set of URLs (by pattern, page type or customer group) and customise their request (User-Agent, headers).
- **Automatic warming** — a cron runner warms pending pages every few minutes; a full cache flush re-queues everything, and saving a product / category / CMS page re-queues just that entity.
- **Built-in & Varnish FPC** — auto-detects the active cache backend and reads real HIT/MISS status from each response.
- **Reports & audit logs** — a per-URL warming report (with CSV / XML export) plus a Cache Flush Log of every requeue event.
- **Built-in FPC test** — a one-click tool that proves caching is actually working for any URL.
- **CLI tools** — `status`, `validate`, `queue:rebuild` and `warm` for automation and CI.
- **Licensed** — activated via the eTechFlow portal; the admin warming pages lock when unlicensed.

1.3 Compatibility matrix

Area	Supported
Magento / Adobe Commerce	2.4.6 – 2.4.9
PHP	8.1 – 8.4
Cache backend	Built-in Full Page Cache & Varnish (auto-detected)
URL sources	XML Sitemap, Products, Categories, CMS, Custom CSV, Layered Navigation
Scheduling	Magento cron (warm every 5 min, rebuild every 30 min — configurable)
Licensing	eTechFlow license-service (SP / HMAC / Bundle keys)

2 · Architecture & How It Works

2.1 The pipeline

Cache Warmer is a self-contained module — **ETechFlow_CacheWarmer** (package `etechflow/module-cache-warmer`). Its work flows through five stages:

1. **Build the queue** — URL providers (sitemap / product / category / CMS / CSV / layered-nav) collect candidate URLs.
2. **Prioritise** — each URL is scored: a page-type base weight, plus best-seller popularity for products, plus any matching rule's priority boost. Higher score = warmed first.
3. **Queue** — scored URLs are de-duplicated and stored as rows in `etechflow_cachewarmer_queue`, one per store / customer-group.

4. **Warm** — the warmer requests each pending URL (concurrently, load-aware) so Magento caches it.
5. **Detect & log** — a cache-status detector reads each response's HIT / MISS / UNCACHEABLE status; every request is written to the request log and surfaced in Reports.

2.2 Key components

Component	Role
UrlProviderPool	Pluggable set of URL sources (sitemap, product, category, CMS, CSV, layered nav).
Prioritizer	Scores each URL = page-type weight + best-seller popularity + rule boost.
RuleResolver	Applies active warming rules — priority boost, custom UA / headers, exclusions.
Warmer	Sends the HTTP requests (concurrency + load throttling) and records results.
DetectorPool	Reads HIT / MISS / UNCACHEABLE from each response (built-in FPC or Varnish).
QueueManager	Upserts, requeues and prunes the queue table; tracks status per row.
Cron jobs	BuildQueue (refresh URLs), RunWarmer (warm pending), Cleanup (prune logs).

2.3 Data & storage

Five tables hold all state: `etechflow_cachewarmer_queue` (the URL queue), **…**`_rule` (warming rules), **…**`_request_log` (every warm request, powering Reports), **…**`_flush_log` (the audit of flush / requeue events) and **…**`_run_log` (warmer run history).

3 · Installation & Setup

3.1 Install & enable

Distributed via the private eTechFlow Composer repository; your licence key doubles as the download token. From your Magento project root:

```
composer config repositories.etechflow composer
https://license-service.etechflow.com/composer

composer config http-basic.license-service.etechflow.com you@example.com SP-XXXX-XXXX-
XXXX-XXXX

composer require etechflow/module-cache-warmer

bin/magento module:enable ETechFlow_CacheWarmer

bin/magento setup:upgrade

bin/magento cache:flush
```

Make sure Magento's **cron** is running (the warmer's automatic warming relies on it) and that the **Full Page Cache** type is enabled.

3.2 Where it lives

Everything sits under the shared **ETECHFLOW** admin menu:

Screen	Path
Dashboard	ETECHFLOW → Cache Warmer → Dashboard
Warming Queue	ETECHFLOW → Cache Warmer → Warming Queue
Warming Rules	ETECHFLOW → Cache Warmer → Warming Rules
Reports	ETECHFLOW → Cache Warmer → Reports
Cache Flush Log	ETECHFLOW → Cache Warmer → Flush Log
FPC Test	ETECHFLOW → Cache Warmer → Test
Configuration	Stores → Configuration → ETECHFLOW → Cache Warmer

First run

After install, open **Configuration**, set **Enable Cache Warmer = Yes**, then go to the **Warming Queue** and click **Rebuild Queue** to populate it. From then on the cron keeps it warm automatically — or click **Warm Now** any time.

4 · The Warming Queue

The Warming Queue is the heart of the module — the live list of URLs to warm and their status. Open it at **ETECHFLOW → Cache Warmer → Warming Queue**.

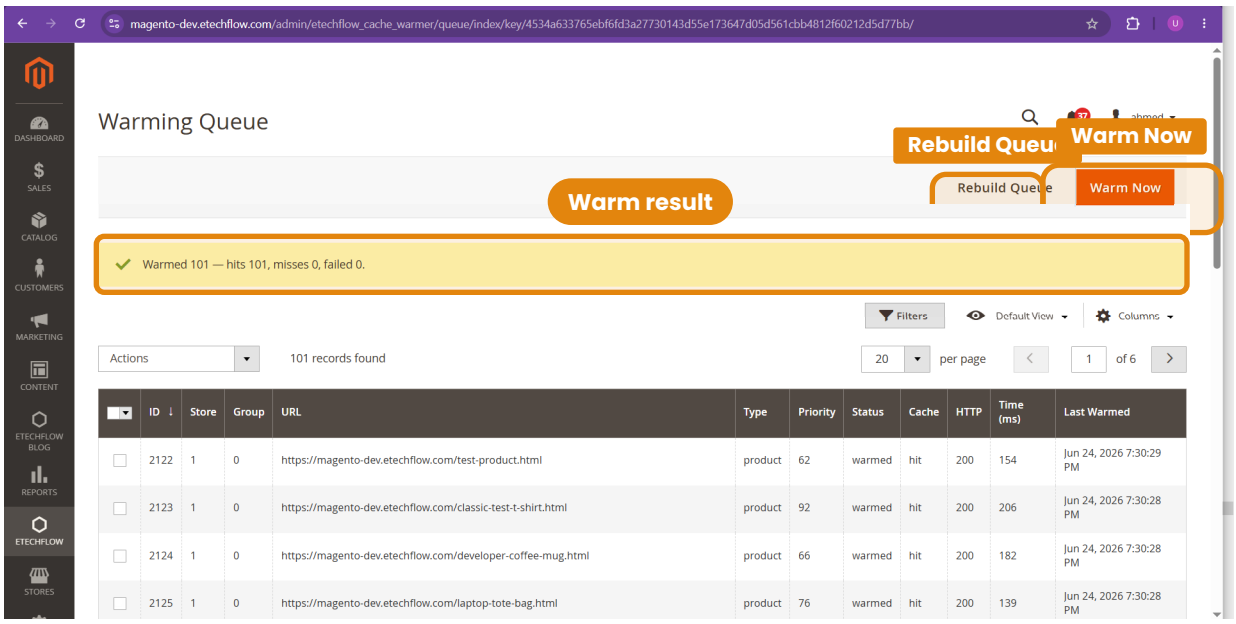

Figure 1 — the Warming Queue: 101 URLs warmed with a 100% cache-hit rate, plus the Rebuild Queue and Warm Now actions.

4.1 Building & warming

- **Rebuild Queue** — clears and re-collects the URL list from every enabled source, re-scoring priorities. Run it after big catalog changes.
- **Warm Now** — immediately processes all **pending** rows: requests each URL, caches it, and records the result. The green banner summarises *"Warmed N — hits X, misses Y, failed Z."*

4.2 The columns

Column	Meaning
ID	Queue row id.
Store / Group	Store view and customer group the row warms (0 = guest / NOT LOGGED IN).

URL	The page to warm.
Type	Page type — product, category, cms, sitemap, layered.
Priority	Warming order — higher is warmed first (see Warming Rules).
Status	pending → warmed (or failed / skipped). pending = queued but not yet fetched.
Cache	Result of the warm — hit (already cached) or miss (rendered & now cached).
HTTP	Response code (200 = OK).
Time (ms)	How long the warm request took.
Last Warmed	Timestamp of the last successful warm.

4.3 Row actions

Tick one or more rows and use the **Actions** menu for bulk operations:

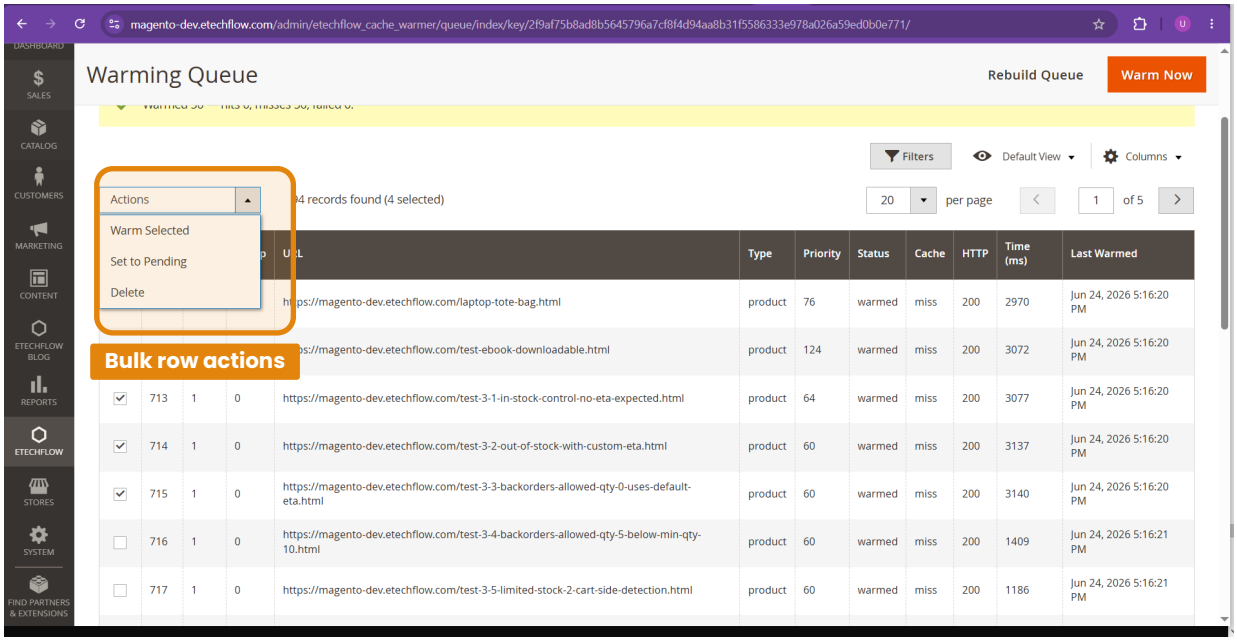

Figure 2 — Queue mass actions: Warm Selected, Set to Pending and Delete.

- **Warm Selected** — warm just the chosen rows now.
- **Set to Pending** — re-queue the chosen rows so they get re-warmed (manually or by the next cron run).
- **Delete** — remove rows from the queue.

5 · Warming Rules & Priority

By default every URL is scored automatically (page-type weight + best-seller popularity). **Warming Rules** let you override that — push a set of URLs higher up the queue so they warm first, and customise how they are requested. Manage them at **ETECHFLOW → Cache Warmer → Warming Rules**.

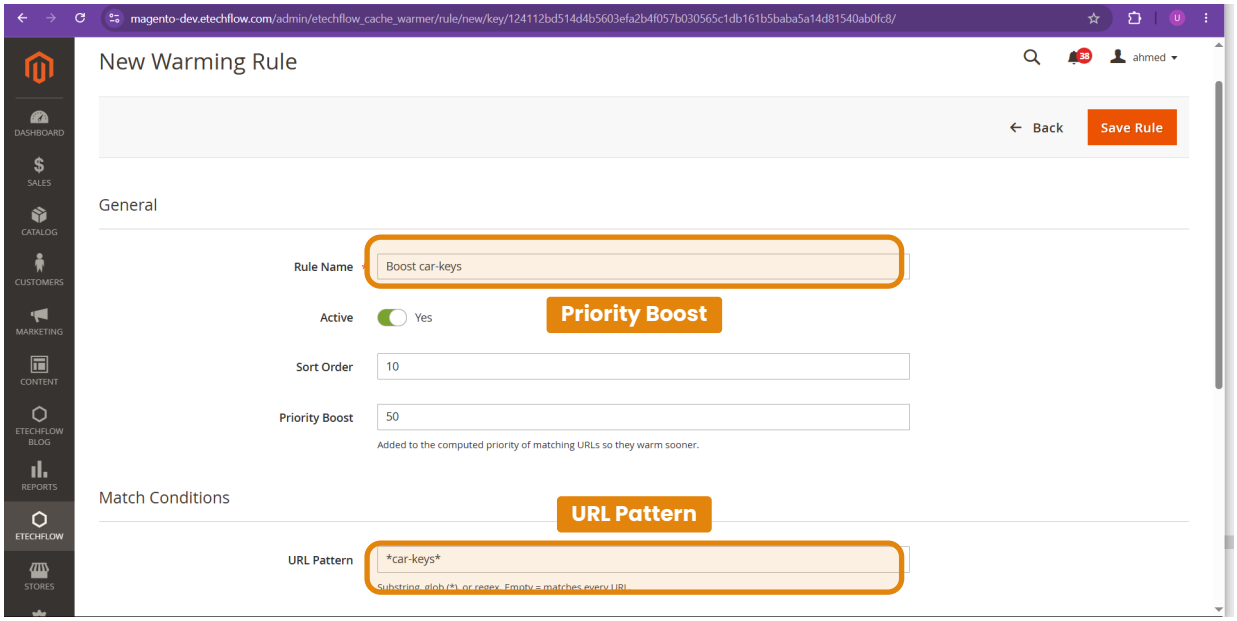

Figure 3 — the New Warming Rule form: General settings and Match Conditions.

5.1 Creating a rule

Each rule has three groups of fields:

- **General** — *Rule Name*, *Active* (on/off), *Sort Order* (which rule wins when several match), and *Priority Boost* (added to the computed priority of every matching URL so it warms sooner).
- **Match Conditions** — *URL Pattern* (substring, glob *, or regex; empty = every URL), *Page Types* and *Customer Groups* to scope the rule.
- **Request Options** — a per-rule *User-Agent Override*, *User-Agent Exclude Regex* (skip when the UA matches) and *Custom Headers* (added to the warm request — e.g. to warm a specific cache variant).

5.2 The priority boost in action

In the example below, a rule with **URL Pattern `*car-keys*`** and **Priority Boost 100** lifts every car-keys page from its base score (category = 100) to **200** — so they sit at the very top of the queue and warm first.

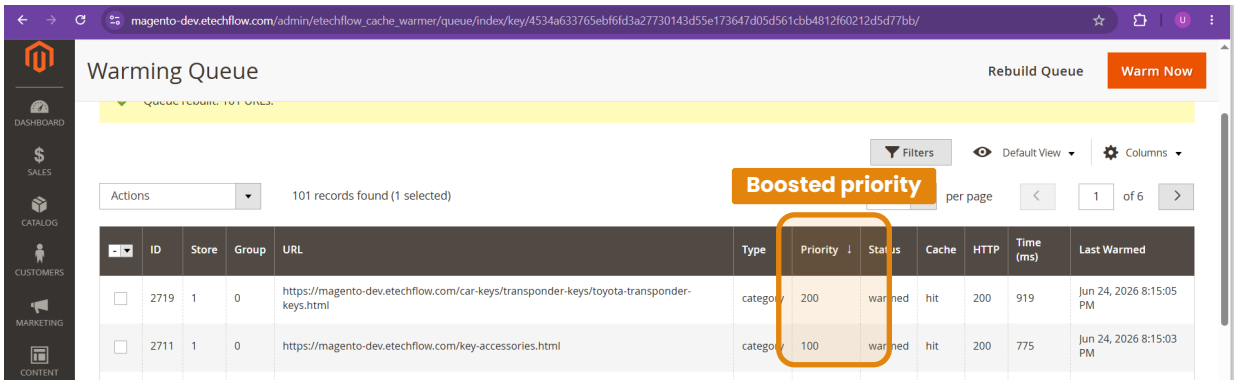

Figure 4 — the queue sorted by Priority: the car-keys category pages are boosted to 200 and warmed first.

How scoring works

Priority = page-type base weight (cms 120, category 100, custom 80, product 60, sitemap 40, layered 20) + best-seller popularity (for products) + the matching rule's Priority Boost. Only **Active** rules apply; when several match, the lowest

Sort Order wins.

6 · Automatic Warming — Triggers & Cron

Cache Warmer keeps the cache warm with no manual clicks. Two things drive it: **triggers** that re-queue pages the moment their cache is cleared, and a **cron** that warms the pending rows.

6.1 Re-queue triggers

- **Auto-Warm on Flush** — a full cache flush (System → Cache Management → Flush Magento Cache) re-queues **every** warmed row back to pending, so the whole site re-warms.
- **Warm After Save** — saving a single product, category or CMS page clears only that entity's cache tag; the module maps the tag back to the affected URLs and re-queues **just those rows**.

Below, editing and saving a product flips only that product's queue row back to **pending** — the rest stay warmed — and the event is recorded in the Cache Flush Log.

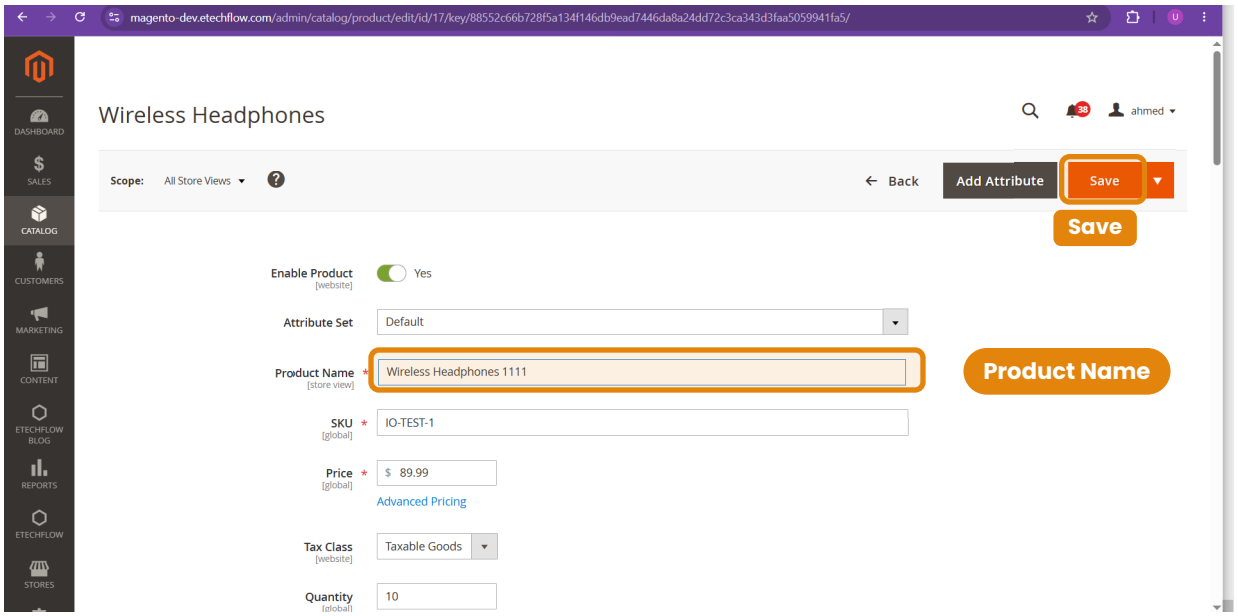

Figure 5 — editing a product (changing its name) and saving it…

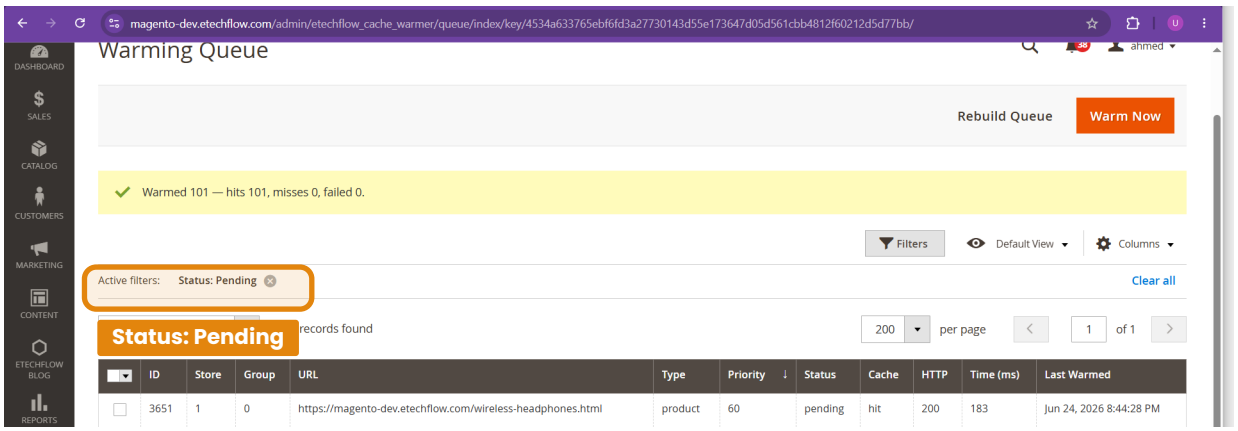

Figure 6 — …flips only that product's row to Pending (queue filtered by Status: Pending); everything else stays warmed.

6.2 The cron jobs

Three scheduled jobs run the automation (all require Magento cron to be running):

Job	Default schedule	What it does
Run Warmer	Every 5 minutes (`*/5 * * * *`)	Warms all pending queue rows — the hands-off engine.
Build Queue	Every 30 minutes (`*/30 * * * *`)	Refreshes the URL list from the enabled providers.
Cleanup	Daily at 03:30	Prunes old logs and releases any stale in-progress locks.

The full automatic loop

Flush cache or save content → the affected rows flip to **pending** → within 5 minutes the **Run Warmer** cron warms them back to a cache HIT — before your next visitor arrives. The warmer schedule is configurable (Configuration → General → Warmer Cron Schedule).

7 · Reports & Flush Logs

7.1 Warming Reports

ETECHFLOW → Cache Warmer → Reports is the per-URL warming history — every warm request the module has made, with its HTTP code, response time, cache HIT/MISS and what triggered it (manual, cron or event). Filter by any column and export the (filtered) data.

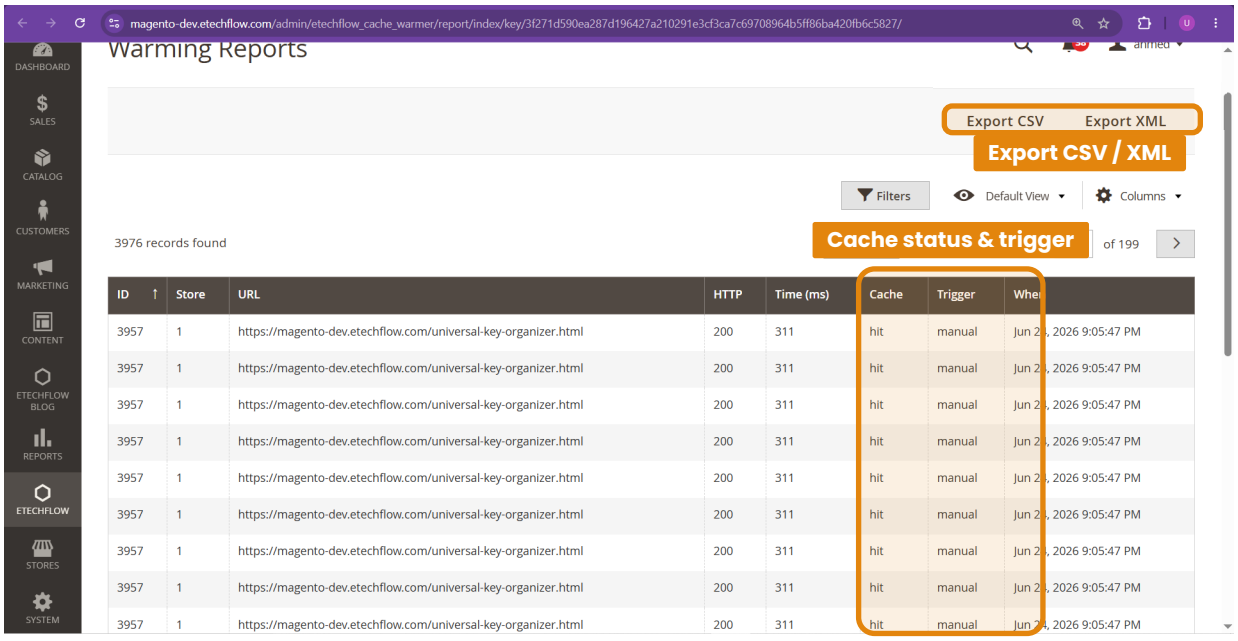

Figure 7 — Warming Reports: per-URL history with HTTP, time, cache status and trigger, plus Export CSV / Export XML.

- **Export CSV** / **Export XML** — download the report for analysis or sharing.
- **Filters** — narrow by cache status (hit/miss), trigger (manual/cron/event), HTTP code or date range.

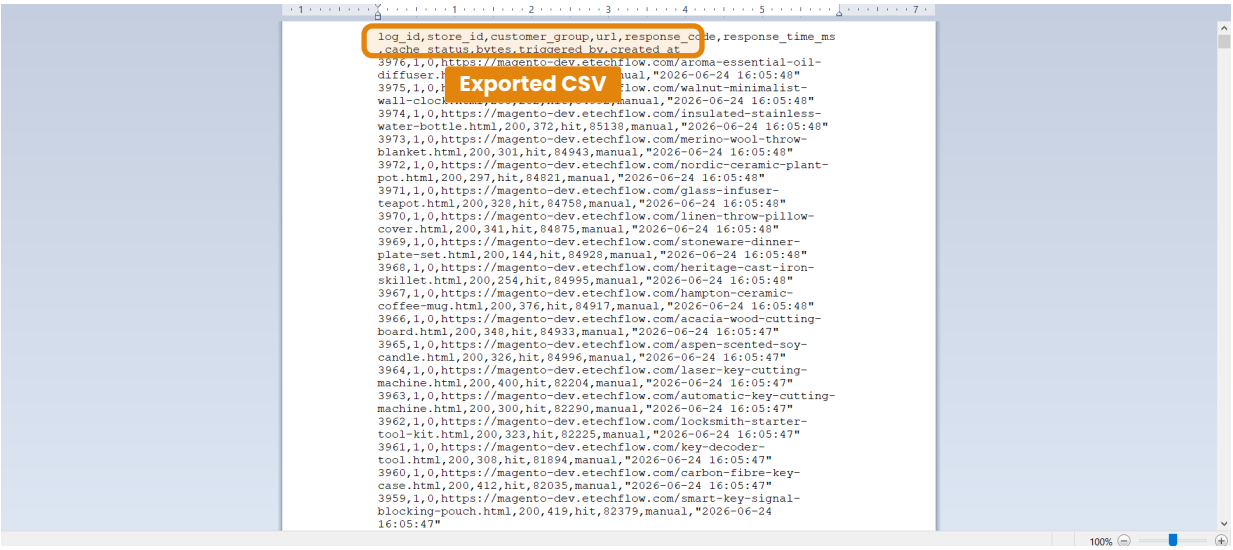

Figure 8 — an exported CSV of the warming report, ready for a spreadsheet.

7.2 Cache Flush Log

ETECHFLOW → Cache Warmer → Flush Log is the audit trail of every re-queue event — proof of the automatic warming at work. Each row shows the **source** (`admin` = a full flush, `event` = a content save), the cache **tags** cleared, how many **URLs were requeued**, and **when**.

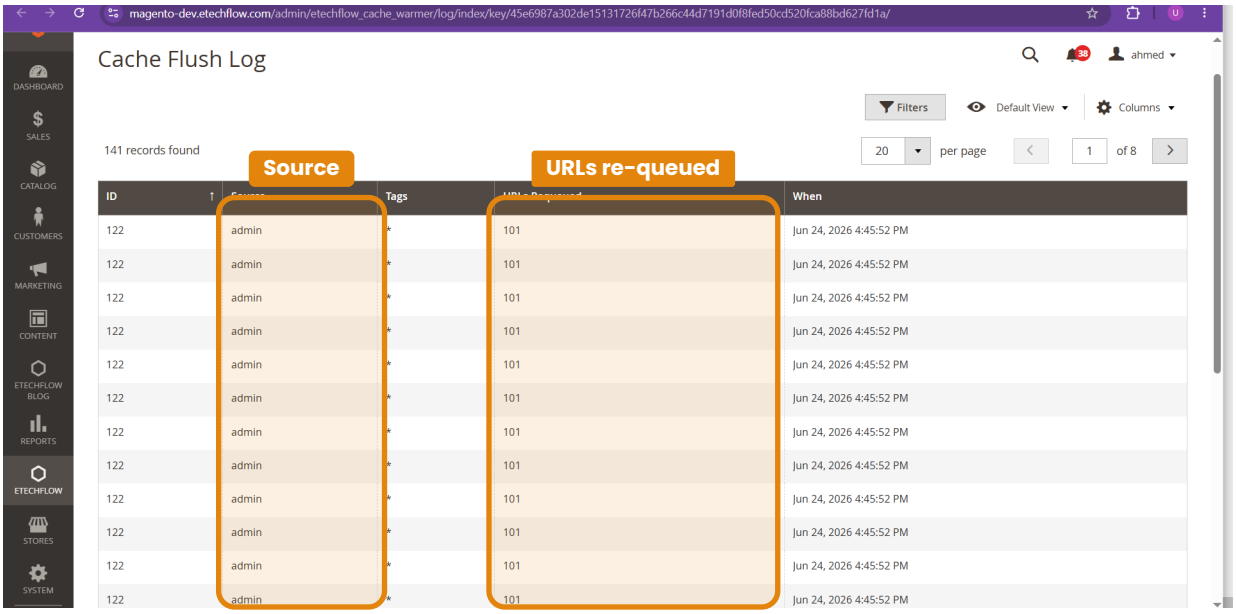

Figure 9 — the Cache Flush Log: every flush / save event and how many URLs it re-queued.

8 · The Built-in FPC Test

Is Full Page Cache actually working? The **FPC Test** tool (**ETECHFLOW → Cache Warmer → Test**) settles it. Enter any storefront URL and click **Run Test** — the tool requests it **twice** and checks that the second response is served from cache (a HIT).

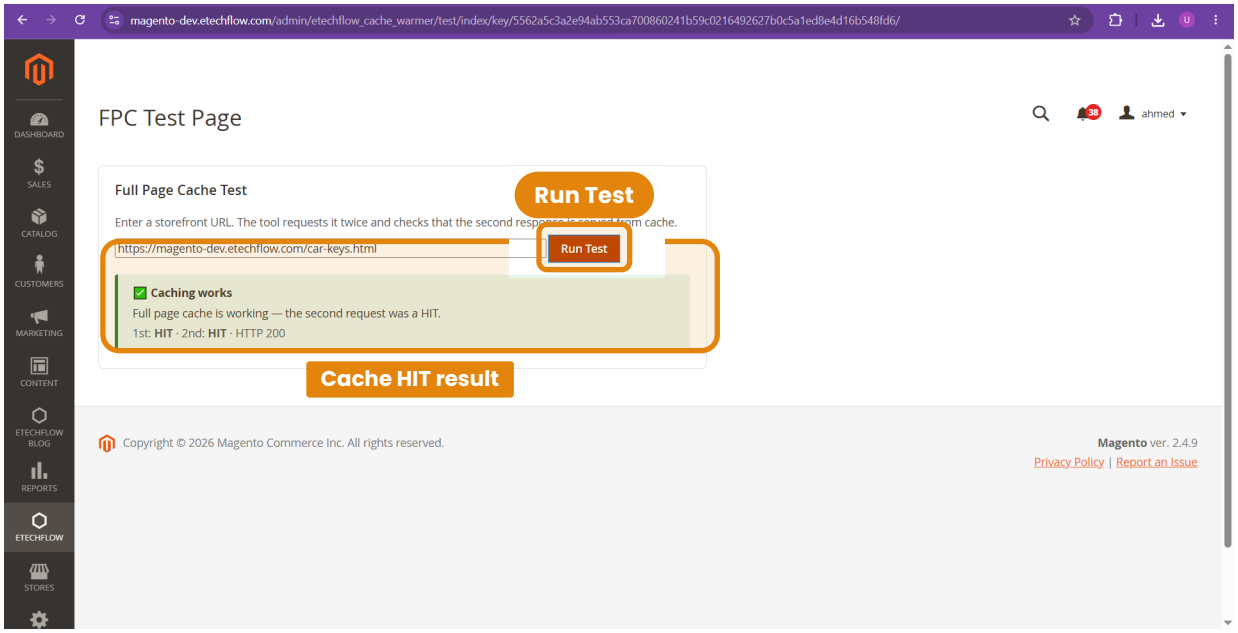

Figure 10 — the FPC Test: a green result confirms "Full page cache is working — the second request was a HIT."

Result	Meaning
Caching works (2nd = HIT)	FPC is working for this URL — exactly what you want.
UNCACHEABLE	A block marked `cacheable="false"` is disabling FPC for the whole page — find and fix it.
Not cached (2nd = MISS)	The URL isn't being cached — check it is a cacheable GET route (no session cookie).
Could not read cache headers	Enable FPC debug headers, or confirm Varnish is in front of the URL.

Tip
Test a URL that returns **HTTP 200**. A 404 / redirect is never cached, so it will always report a MISS — that's correct behaviour, not a caching fault.

9 · Command-Line (CLI) Tools

Four `bin/magento` commands cover automation, cron and CI use:

Command	Purpose
`etechflow:cw:status`	Show queue counts and cache hit/miss efficiency.
`etechflow:cw:validate`	Verify FPC works by requesting a URL twice (the CLI FPC test).
`etechflow:cw:queue:rebuild`	Clear and rebuild the URL queue.
`etechflow:cw:warm`	Warm the cache by processing all pending queue rows now.

Typical output of `status` and `warm`:

```
$ bin/magento etechflow:cw:status
```

```
Cache Warmer status

  Total queued : 101    Warmed : 101    Failed : 0

  Cache hits   : 101    Cache misses : 0

  Hit rate     : 100%   Avg response : 371 ms


$ bin/magento etechflow:cw:queue:rebuild

Queued 101 URLs.

$ bin/magento etechflow:cw:warm

Processed 101 — warmed 101, failed 0, hits 101, misses 0.
```

10 · Configuration Reference

All options live at **Stores → Configuration → ETECHFLOW → Cache Warmer** (`etechflow_cache_warmer`). A **Module Status** banner at the top tells you, at a glance, whether the licence is valid and whether warming is switched on.

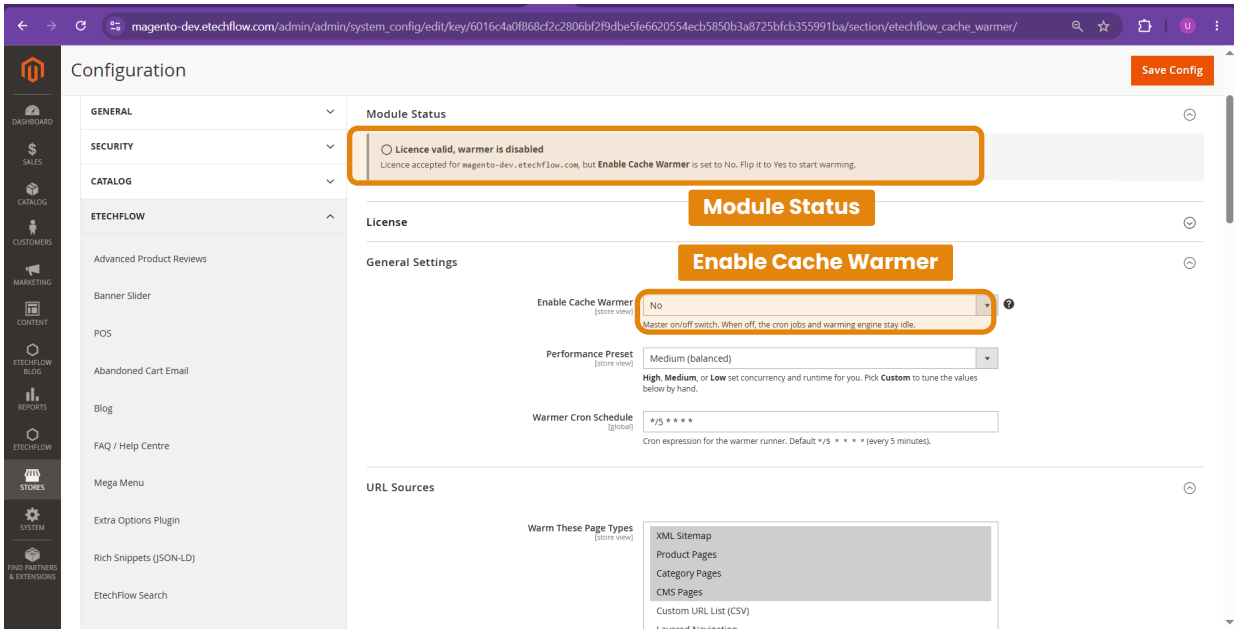

Figure 11 — Configuration: the Module Status banner, General Settings and URL Sources.

10.1 General & URL Sources

Setting	Default	Purpose
Enable Cache Warmer	No	Master on/off switch. When off, the cron jobs and warming engine stay idle.
Performance Preset	Medium	High / Medium / Low set concurrency & runtime for you; Custom unlocks the values below.

Warmer Cron Schedule	*/5 * * * *	Cron expression for the warmer runner (default every 5 minutes).
Warm These Page Types	Sitemap, Product, Category, CMS	Which URL sources to crawl (also Custom URL List (CSV) and Layered Navigation).
Max URLs	50000	Hard cap on queue size per rebuild.

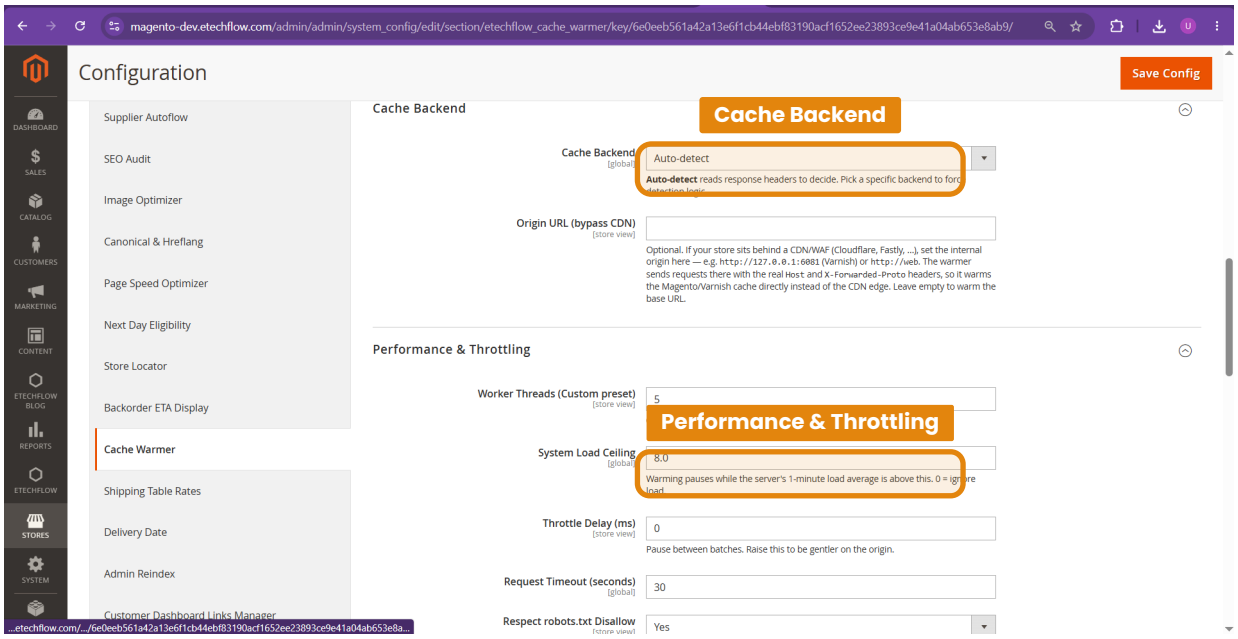

Figure 12 — Cache Backend and Performance & Throttling settings.

10.2 Cache Backend & Performance

Setting	Default	Purpose
Cache Backend	Auto-detect	Built-in FPC or Varnish. Auto-detect reads response headers; force one to override.
Origin URL (bypass CSP)	(empty)	Optional internal origin to warm against, bypassing an edge CDN / proxy.
Worker Threads	5	Concurrent warm requests (Custom preset).
System Load Ceiling	8.0	Pause warming if the server's 1-minute load average exceeds this (0 = ignore).
Throttle Delay (ms)	0	Pause between requests to be gentler on the origin.
Request Timeout (seconds)	30	Per-request timeout.
Respect robots.txt Disallow	Yes	Skip URLs disallowed by robots.txt.

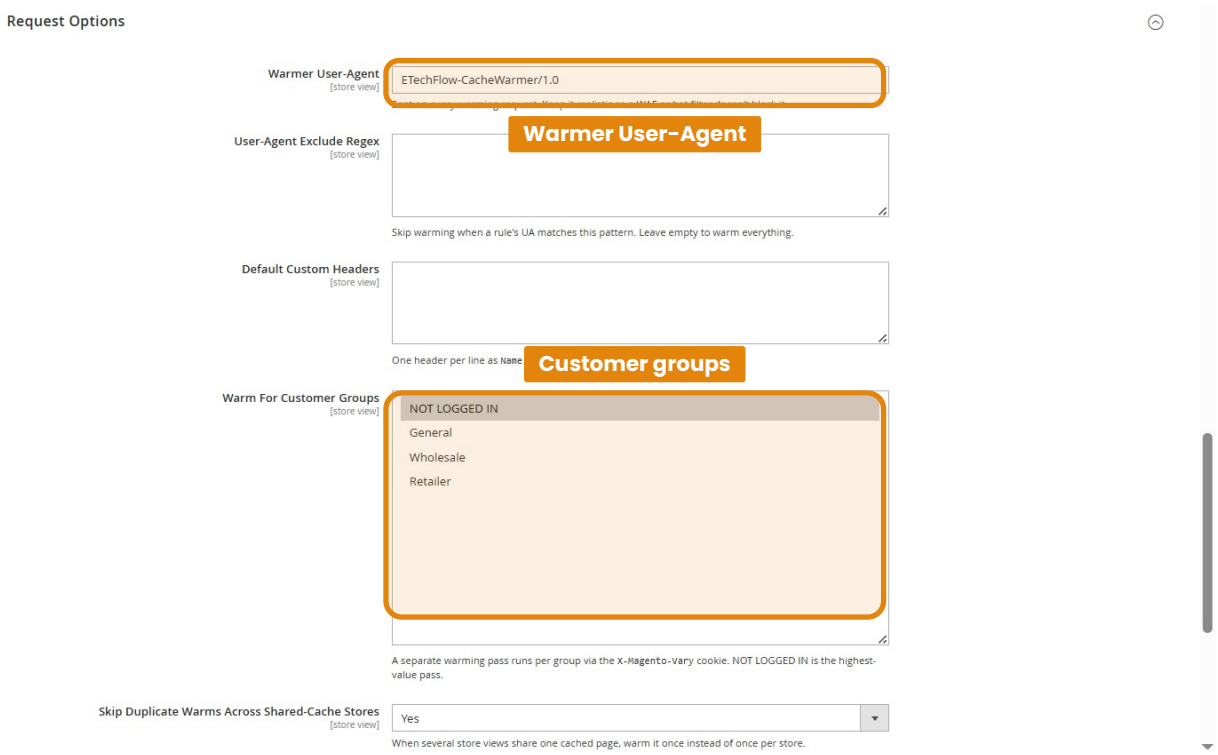

Figure 13 — Request Options: User-Agent, custom headers and customer-group warming.

10.3 Request Options, Triggers & Logging

Setting	Default	Purpose
Warmer User-Agent	ETechFlow-CacheWarmer/1.0	Sent on every warm request — keep it realistic so a WAF / bot filter doesn't block it.
User-Agent Exclude Regex	(empty)	Skip warming when a rule's UA matches this pattern.
Default Custom Headers	(empty)	Headers added to every warm request (Name: Value per line). Rule headers override these.
Warm For Customer Groups	NOT LOGGED IN	A separate warming pass runs per group via the X-Magento-Vary cookie.
Skip Duplicate Warms (shared cache)	Yes	If several store views share one cached page, warm it once instead of per store.
Auto-Warm on Flush	Yes	Re-queue everything when the cache is fully flushed.
Warm After Save	Yes	Re-queue the affected URLs when a product / category / CMS page is saved.
Enable Logging / Debug / Retention	On / Off / 30 days	Request logging, verbose debug output, and how long to keep logs.

11 · Licensing & Activation

Cache Warmer is activated with a key from the eTechFlow portal, validated live against `license-service.etechflow.com` and bound to your licensed domain.

6. Obtain a key (**SP-XXXX-XXXX-XXXX-XXXX**) from your eTechFlow account — it is also your Composer download token.
7. Go to **Stores → Configuration → ETECHFLOW → Cache Warmer → License**, paste the key, and **Save Config**.
8. The **Module Status** banner confirms the licence — e.g. *"Licence valid"* — and reminds you to set Enable Cache Warmer = Yes.

Enforcement
When the licence is missing, suspended, expired or bound to a different domain, the admin Cache Warmer pages lock to the licence gate. The Configuration and License screens always stay reachable so you can re-activate.

12 · Troubleshooting / FAQ

Every page reports MISS, even on a second warm.

The store isn't caching. Run the FPC Test on a URL — if it says **UNCACHEABLE**, a block somewhere is marked `cacheable="false"`, which disables FPC for the whole page (a single such block in the `default` layout disables it site-wide). Find and remove it, flush cache, and re-test.

Queue rows are stuck on Pending and never warm by themselves.

Automatic warming needs **Magento cron** running. Check `bin/magento cron:install` / your cron container, and confirm the `etechflow_cache_warmer_run_warmer` job shows recent successful runs. Until then, use Warm Now or `bin/magento etechflow:cw:warm`.

The whole queue went back to Pending after I flushed cache.

That's the Auto-Warm on Flush trigger working as designed — a full flush re-queues everything so it gets re-warmed. The cron (or Warm Now) brings it back to warmed.

The FPC Test shows MISS / HTTP 404.

The URL you entered doesn't exist (404) or isn't a cacheable route. 404s and redirects are never cached. Re-test with a real product / category / CMS URL that returns HTTP 200.

Warm Now says "Processed 0".

There are no pending rows — everything is already warmed. Click Rebuild Queue (or wait for a flush / save) to create pending work, then warm.

Does it work with Varnish?

Yes. The Cache Backend setting auto-detects built-in FPC or Varnish and reads each response's real HIT/MISS status; you can also force a backend.

How do I make some pages warm first?

Create a Warming Rule with a URL Pattern that matches them and a positive Priority Boost — matching URLs jump up the queue. See section 5.