

SUPER SPEED

User Guide

Speed optimization module for PrestaShop

Compatible with PrestaShop 1.7.x to PrestaShop 9.x | by PrestaHero

This guide helps you make your online shop faster without needing technical skills. It explains what each part of the module does, which settings matter, and which setup to use for your type of website.

> **How to use this guide**
>
> **In a hurry?** Read Chapter 3 (Quick start). It takes about 10 minutes and gives you most of the benefit.
>
> **Want the best setup for your shop?** Jump to Part 3 and pick the recipe that matches your website.
>
> **Something looks wrong?** Go to Part 4 (Troubleshooting).

Table of Contents

1. What Super Speed does *4*

2. Before you install *5*

3. Quick start in 10 minutes *6*

4. Dashboard *8*

5. Page cache *8*

5.1 Turn it on and choose what to cache 10

5.2 How the cache stays fresh 10

5.3 Different visitors, different copies 11

5.4 URL rules tab 12

5.5 Dynamic content and Live JavaScript 12

5.6 Cache records and logs 13

6. Cloudflare (optional) *14*

7. Server cache and minification *14*

8. Browser cache and compression *15*

9. Image optimization *16*

9.1 Choose how to optimize 18

9.2 Bulk and automatic optimization 18

9.3 Upload to optimize and Browse images 18

9.4 Image cleaner 19

9.5 Lazy load 19

10. Database cleanup *19*

11. System Analytics *20*

12. Automation (cron job) *21*

13. Choosing your recipe *23*

Recipe A: Catalogue / showcase site 23

Recipe B: Standard online store 24

Recipe C: Multi-currency / multi-country store 25

Recipe D: Group pricing / B2B store 25

Recipe E: High-traffic store with Cloudflare 26

Recipe F: Content / blog-driven site 26

Recipe G: Limited hosting resources 27

14. Comparison table *28*

15. After you change settings *29*

16. Troubleshooting *29*

17. Turning things off *30*

18. FAQ 31

Appendix: cache status codes 31

PART 1

Getting started

1. What Super Speed does

Every time someone opens a page on your shop, PrestaShop normally builds that page from scratch: it asks the database for products, prices and text, assembles the layout and sends it out. This takes time, and it takes more time the more visitors you have.

Super Speed reduces that work. Its most important feature is **page cache**, which saves a finished copy of each page and hands it out to the next visitors instantly. On top of that, it makes files smaller, makes images lighter and keeps your database tidy.

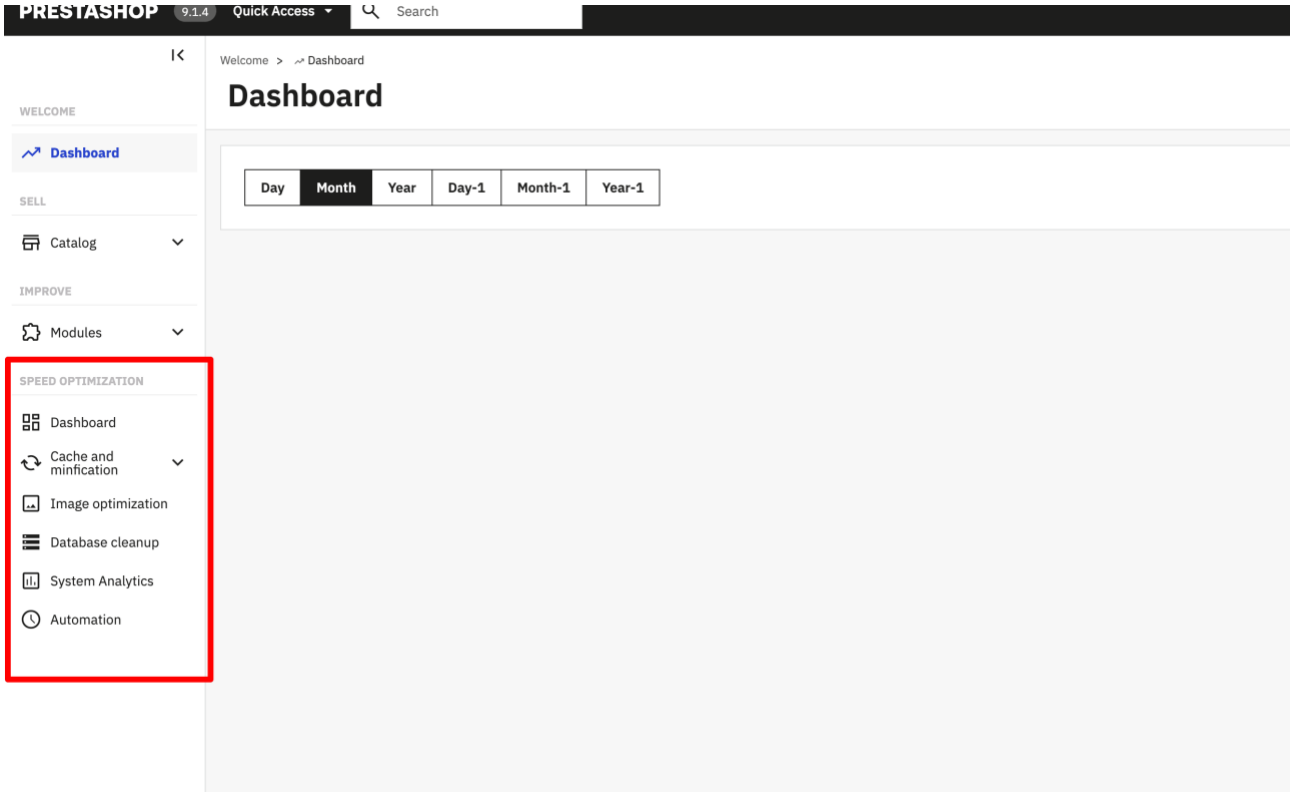
Key terms in plain English

Term	What it means for you
Page cache	A saved, ready-made copy of a page. The first visitor builds it; later visitors get the saved copy.
Minify	Removing spaces and comments from your page code (HTML, CSS, JavaScript) so files are smaller. Visitors see no difference.
Compression	Packing files into a smaller size while they travel to the visitor, like a zip file. Brotli and Gzip are two ways of doing it.
Browser cache	Telling the visitor's browser to keep your logo, styles and images for a while so it does not download them again on every page.
Image optimization	Making image files smaller while keeping them looking good.
WebP	A modern image format that is usually smaller than JPG at similar quality. Most current browsers support it.
Lazy load	Loading images only when the visitor scrolls near them, so the top of the page appears sooner.
Cron job	A task your server runs on a schedule (for example, every hour) to do housekeeping automatically.
Cloudflare / CDN	An outside service that keeps copies of your pages on servers around the world. Optional.

Where to find everything

After installation, a new menu called **Speed Optimization** appears in the Back Office:

Menu item	What it is for
Dashboard	Your speed overview, a checklist, and one-click Auto configuration.
Cache and minification › Page cache	The main cache settings, Cloudflare, dynamic content and cache records.
Cache and minification › Server cache and minification	Smarty cache, server cache, and minify HTML, CSS and JavaScript.
Cache and minification › Browser cache and compression	How long browsers keep your files, and file compression.
Image optimization	Compress images, convert to WebP, remove unused images, lazy load.
Database cleanup	Delete old logs, carts, expired discounts and similar clutter.
System Analytics	Find what is slowing your shop down, including slow modules.
Automation	Set up the scheduled housekeeping (cron job).



2. Before you install

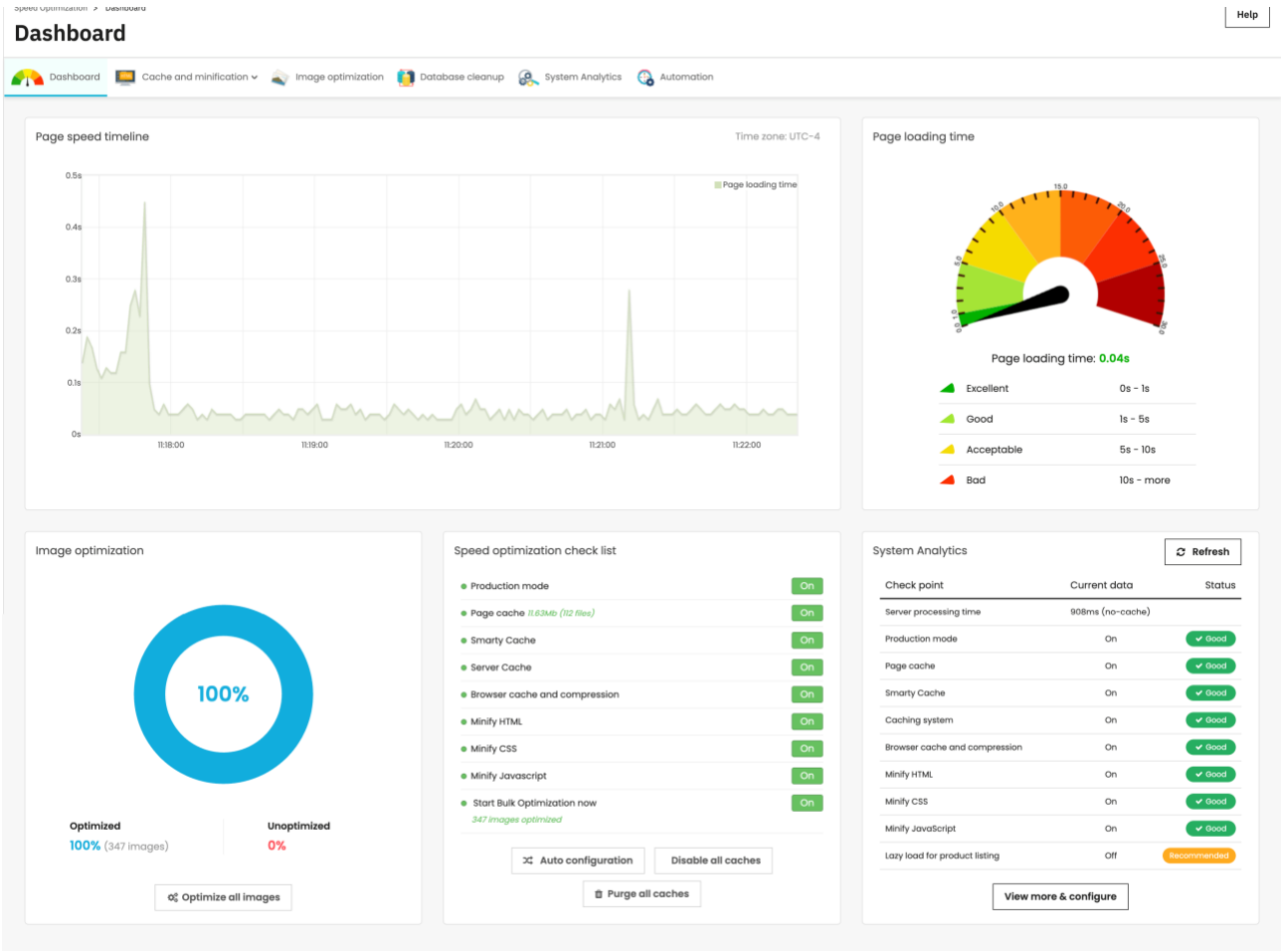
- **PrestaShop version:** 1.7 or newer (the module supports up to PrestaShop 9.x).
- **One licence per website.** If you run several domains, you need a licence for each.

- **Do not install it next to similar modules.** Super Speed cannot be installed together with the PrestaHero **Page Cache** module (`ets_pagecache`) or **Image Optimization** module (`ets_imagecompressor`). Uninstall those first.
- **Apache vs. Nginx:** the "Browser cache and compression" feature works through the `.htaccess` file, which only Apache uses. On Nginx, ask your hosting provider to set up the equivalent. Everything else works on both.
- **Make a backup** of your files and database before you install. The module adds small patches to a few PrestaShop core files so it can serve cached pages early. If another module changes the same files, the Dashboard will show a warning (see **Troubleshooting**).

3. Quick start in 10 minutes

These steps give you a solid setup for most shops. You can fine-tune later using the recipes in Part 3.

1. **Open the Dashboard.** Go to **Speed Optimization › Dashboard**. You will see a speed gauge, a checklist and three buttons: **Auto configuration**, **Disable all caches** and **Purge all caches**.
2. **Click Auto configuration.** A panel lists the optimizations it can switch on: Production mode, Page cache, Smarty Cache, Server Cache, Browser cache and compression, Minify HTML, Minify CSS and Minify JavaScript. Confirm and wait for the "Everything is done" message.
3. **Optimize your existing images (recommended).** The same panel offers **Start Bulk Optimization**. It works through your images in the background. You can pause or stop it at any time.
4. **Set up the cron job.** Go to **Speed Optimization › Automation** and copy the command into your server's crontab (see Chapter 12). This keeps the cache and database tidy automatically.
5. **Check your shop as a visitor.** Open your shop in a private browser window. Browse the home page, a category, a product, then add a product to the cart and go to checkout. Everything should look and work as normal.
6. **Measure.** Run your home page through a speed test (for example Google PageSpeed Insights) and compare with your earlier result. Test the same page twice: the first visit builds the cache, the second shows the real benefit.



Good to know about Auto configuration

It turns on PrestaShop **production mode** (it switches debug mode off). This is right for a live shop, but if you are still developing your site, leave that option off.

Auto configuration also rewrites the cache and compression part of your `.htaccess` file. If you added your own rules, keep a copy of them.

PART 2

Features, one by one

Each chapter follows the same pattern: what the feature does, which settings matter and what to watch out for. Settings are shown with their default value, which is what you get after installation.

4. Dashboard

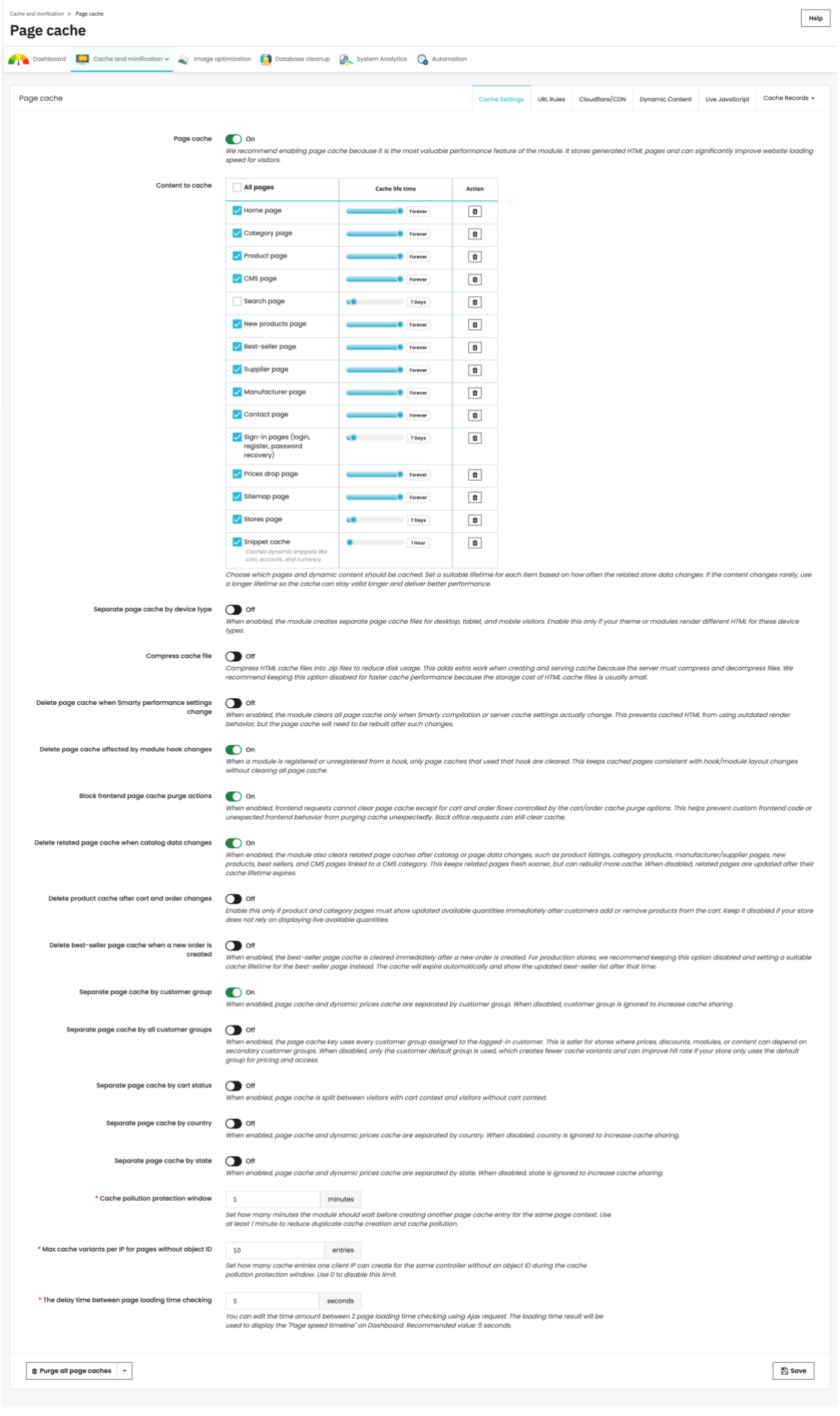
The Dashboard is your control tower. It shows:

- **Page speed timeline:** a live chart of how fast your home page responds. It is refreshed every 5 seconds while the Dashboard is open (you can change that interval in the Page cache settings).
- **Checklist:** items such as Production mode, Page cache, Smarty Cache, Browser cache, Minify and Lazy load, each marked as good or recommended.
- **Image optimization status:** how many images are optimized and how much space you saved.
- **Warnings:** for example, when Cloudflare login details stopped working, or when debug-style tracking options are left on.

Button	What it does
Auto configuration	Switches on the recommended optimizations in one go (see Chapter 3).
Disable all caches	Switches off page cache, Smarty and server cache, minify, and browser cache and compression in one go. Useful if you suspect the module is behind a problem and want to rule it out quickly. Image optimization and lazy load are not changed. It also clears your “Content to cache” selection, so after switching things back on, tick your pages again (or click Auto configuration).
Purge all caches	Deletes all saved page copies so every page is rebuilt on its next visit.

5. Page cache

Speed Optimization › Cache and minification › Page cache. This is the heart of the module and gives the biggest speed gain. The page has these tabs: **Cache settings**, **URL rules**, **Cloudflare/CDN**, **Dynamic content**, **Live JavaScript** and **Cache records**.


Cache and minification > Page cache
Page cache
Help
Dashboard
Cache and minification
Image optimization
Database cleanup
System Analytics
Automation
Page cache
Cache Settings
URL Rules
Cloudflare/CDN
Dynamic Content
Live JavaScript
Cache Records
Page cache
On
We recommend enabling page cache because it is the most valuable performance feature of the module. It stores generated HTML pages and can significantly improve website loading speed for visitors.
Content to cache
All pages
Cache life time
Action
Home page
Forever
Category page
Forever
Product page
Forever
CMS page
Forever
Search page
7 Days
New products page
Forever
Best-seller page
Forever
Supplier page
Forever
Manufacturer page
Forever
Contact page
Forever
Sign-in pages (login, register, password recovery)
7 Days
Prices drop page
Forever
Sitemap page
Forever
Stores page
7 Days
Snippet cache
Caches dynamic snippets like cart, account, and currency.
1 Hour
Choose which pages and dynamic content should be cached. Set a suitable lifetime for each item based on how often the related store data changes. If the content changes rarely, use a longer lifetime so the cache can stay valid longer and deliver better performance.
Separate page cache by device type
Off
When enabled, the module creates separate page cache files for desktop, tablet, and mobile visitors. Enable this only if your theme or modules render different HTML for these device types.
Compress cache file
Off
Compress HTML cache files into .zip files to reduce disk usage. This adds extra work when creating and serving cache because the server must compress and decompress files. We recommend keeping this option disabled for faster cache performance because the storage cost of HTML cache files is usually small.
Delete page cache when Smarty performance settings change
Off
When enabled, the module clears all page cache only when Smarty compilation or server cache settings actually change. This prevents cached HTML from using outdated render behavior, but the page cache will need to be rebuilt after such changes.
Delete page cache affected by module hook changes
On
When a module is registered or unregistered from a hook, only page caches that used that hook are cleared. This keeps cached pages consistent with hook/module layout changes without clearing all page cache.
Block frontend page cache purge actions
On
When enabled, frontend requests cannot clear page cache except for cart and order flows controlled by the cart/order cache purge options. This helps prevent custom frontend code or unexpected frontend behavior from purging cache unexpectedly. Back office requests can still clear cache.
Delete related page cache when catalog data changes
On
When enabled, the module also clears related page caches after catalog or page data changes, such as product listings, category products, manufacturer/supplier pages, new products, best sellers, and CMS pages linked to a CMS category. This keeps related pages fresh sooner, but can rebuild more cache. When disabled, related pages are updated after their cache lifetime expires.
Delete product cache after cart and order changes
Off
Enable this only if product and category pages must show updated available quantities immediately after customers add or remove products from the cart. Keep it disabled if your store does not rely on displaying live available quantities.
Delete best-seller page cache when a new order is created
Off
When enabled, the best-seller page cache is cleared immediately after a new order is created. For production stores, we recommend keeping this option disabled and setting a suitable cache lifetime for the best-seller page instead. The cache will expire automatically and show the updated best-seller list after that time.
Separate page cache by customer group
On
When enabled, page cache and dynamic prices cache are separated by customer group. When disabled, customer group is ignored to increase cache sharing.
Separate page cache by all customer groups
Off
When enabled, the page cache key uses every customer group assigned to the logged-in customer. This is safer for stores where prices, discounts, modules, or content can depend on secondary customer groups. When disabled, only the customer default group is used, which creates fewer cache variants and can improve hit rate if your store only uses the default group for pricing and access.
Separate page cache by cart status
Off
When enabled, page cache is split between visitors with cart context and visitors without cart context.
Separate page cache by country
Off
When enabled, page cache and dynamic prices cache are separated by country. When disabled, country is ignored to increase cache sharing.
Separate page cache by state
Off
When enabled, page cache and dynamic prices cache are separated by state. When disabled, state is ignored to increase cache sharing.
* Cache pollution protection window
1
minutes
Set how many minutes the module should wait before creating another page cache entry for the same page context. Use at least 1 minute to reduce duplicate cache creation and cache pollution.
* Max cache variants per IP for pages without object ID
10
entries
Set how many cache entries one client IP can create for the same controller without an object ID during the cache pollution protection window. Use 0 to disable this limit.
* The delay time between page loading time checking
5
seconds
You can edit the time amount between 2 page loading time checking using Ajax request. The loading time result will be used to display the "Page speed timeline" on Dashboard. Recommended value: 5 seconds.
Purge all page caches
Save

5.1 Turn it on and choose what to cache

In **Cache settings**, switch **Page cache** on, then tick the pages in **Content to cache**. Each page type has its own lifetime slider, which is how long a saved copy is kept before it is rebuilt.

Page type	Default lifetime	Notes
Home page, Category pages	5 days	These change most often (new products, banners).
Product pages, CMS pages	15 days	Updated automatically when you save the product or page.
Search page	1 day	Search results vary a lot, so keep this short.
New products, Best sellers, Suppliers, Manufacturers, Contact, Sign-in pages, Prices drop, Sitemap, Stores	7 days	
Checkout, Blog, Help center, Helpdesk, Product feed, Collections	7 days (Product feed: 1 day)	Only shown when the matching module is installed.
Snippet cache	1 hour	Saves small live parts of a page, such as the cart box. Range 1–24 hours.
Dynamic prices cache	1 day	Only used with Cloudflare full HTML mode (Chapter 6).

For page types measured in days, the slider goes from 1 to 30 days, and the last position means **Forever** (never expires on its own).

> **Tip: how long should a lifetime be?**
> Longer is faster, because pages are rebuilt less often. Because the module refreshes pages automatically when you edit things in the Back Office (see 5.2), long lifetimes are safe for content you manage there. Use shorter lifetimes if your catalogue is changed by an import, ERP or another system that PrestaShop does not know about.

5.2 How the cache stays fresh

You normally never need to clear the cache by hand. The module deletes the affected pages automatically when you:

- save, add or delete a product, category, CMS page, manufacturer, supplier or store;
- change product images or stock;
- place a new order (affects stock and optionally the best-seller page);

- register or unregister a module on a hook (only the pages using that hook are cleared);
- change Smarty, minify or other performance settings.

If you want to refresh by hand: use **Purge all page caches** (bottom of the Cache settings tab, or **Purge all caches** on the Dashboard). When you edit a product, category, CMS page, manufacturer or supplier, you will also find a button to clear just that page's cache.

Setting (Cache settings tab)	Default	In plain words
Delete page cache when Smarty performance settings change	On	Rebuild pages after changes to PrestaShop's Smarty settings.
Delete page cache affected by module hook changes	On	Rebuild only the pages that a module change touches.
Delete related page cache when catalog data changes	On	When a product changes, also refresh listings that show it (category, new products, best sellers). Turn off to rely on lifetimes instead.
Delete product cache after cart and order changes	Off	Turn on only if you show **live stock quantities** on product or category pages.
Delete best-seller page cache when a new order is created	Off	Keep off on busy shops; the 7-day lifetime refreshes the list.
Block frontend page cache purge actions	On	Stops visitors' requests (or faulty front-end code) from clearing your cache. Back Office requests can still clear it.

The module also has switches to refresh pages when you edit **Cross Selling PRO (ets_crosssell)**, **Home Products PRO (ets_homecategories)**, **Mega Menu PRO (ets_megamenu)** or **Custom fields & tabs on product page (ets_extraproducttabs) modules** (all On by default).

5.3 Different visitors, different copies

Language and currency always get their own cached copies. For other differences, you decide. Every extra option creates more copies: that is more accurate but also uses more disk space and means fewer visitors get a ready copy. **Switch an option on only when your pages really look different for that reason.**

Separate page cache by…	Default	Turn on when…
Device type	Off	Your theme sends different HTML to desktop, tablet and mobile (not just a responsive layout).
Customer group	Off	Prices, discounts or content depend on the customer group (for example wholesale vs. retail). A sub-option, **all customer groups**, is for customers in several groups.

Separate page cache by…	Default	Turn on when…
Login status	Off	Logged-in customers see different page content than guests (for example, prices hidden for guests).
Cart status	Off	Pages show different content when the cart is not empty and that content is not loaded live.
Country	Off	Prices, taxes or messages depend on the visitor's country.
State	Off	Taxes depend on the state (for example, US sales tax).

When none of the group or login options are on, customer-specific parts of the page (such as the "Sign in" link, the customer's name and the cart box) are loaded separately after the page appears, so one saved copy can safely serve everybody.

Two protection settings are best left alone: **Cache pollution protection window** (1 minute) and **Max cache variants per IP for pages without object ID** (10). They stop one visitor or bot from flooding your disk with copies.

Compress cache file (Off) zips each saved page to save disk space but costs a little speed. Leave it off unless disk space is really tight.

5.4 URL rules tab

Here you control which web addresses are cached.

- **URLs not cached:** one address fragment per line. Any address containing it skips the cache. The default list already excludes the cart, checkout, account, addresses, order history and similar private pages. You normally only add things here (for example `/my-special-page`), not remove them.
- **Query parameter handling:** many links carry extra tracking codes (for example `?utm_source=...`). Three modes: **Normalize and bypass unknown URLs** (default: tracking codes are ignored, and any other unknown parameter skips the cache, which is the safest), **Normalize and ignore unknown parameters** (unknown parameters are ignored, which gives more cache hits), and **Do not normalize** (every address is saved as-is).
- **Cacheable extra query parameters** and **Custom ignored query parameters:** add your own parameter names, one per line.

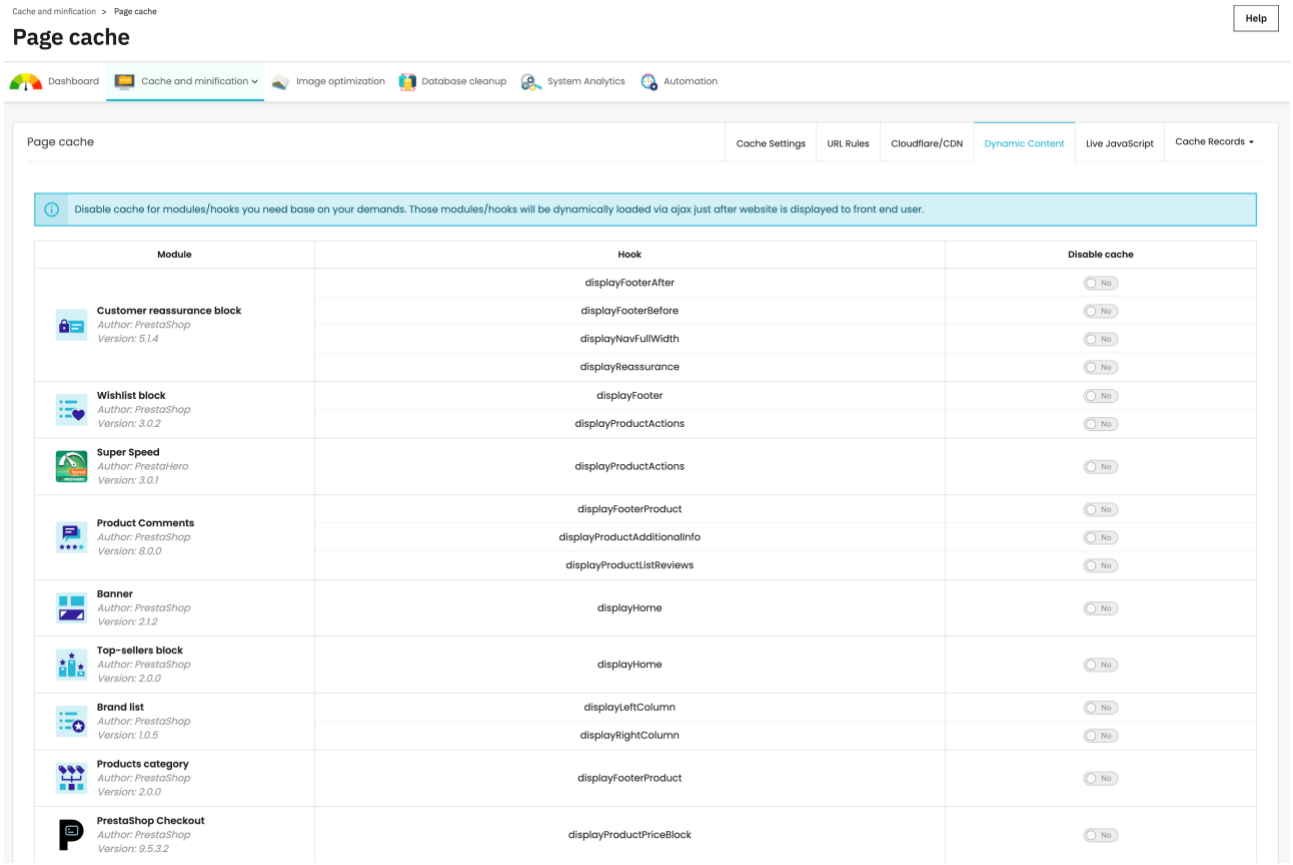
Tracking parameters such as `utm_*`, `gclid`, `fbclid` and `msclkid` are ignored by default, so your ad campaigns do not create thousands of separate copies.

5.5 Dynamic content and Live JavaScript

Some parts of a page must stay live even when the page itself is cached: the cart, the customer's name, the currency selector, recently viewed products. Super Speed handles the common

PrestaShop modules for these automatically and loads them in the background after the page appears.

- **Dynamic content tab:** if your theme has a module that shows live information (for example a stock countdown or personalized block) and it looks stale on cached pages, add that module and its hook here so it is loaded live.
- **Live JavaScript tab:** custom JavaScript that runs after live content has loaded. Enter the code **without** `<script>` tags. Be careful: a mistake here can cause errors across your shop.



5.6 Cache records and logs

The **Cache records** tab shows what is stored and why some pages are not: **Cached pages**, **Cached snippets**, **Cached prices**, **Cache skip logs** and **Cache purge logs**. Hit counters and logs are switched off by default because they add a little extra work. Turn them on while you investigate a problem, then switch them off again (the Dashboard reminds you if you forget).

> **Tip: check whether a page came from the cache**
> Open the page in your browser, open the developer tools (F12), choose the Network tab, reload, click the first request and look at the response header called `X-SS`. **HIT** means the page came from the cache. **MISS** means it was built now and saved. The Appendix explains the other codes.

6. Cloudflare (optional)

If your domain runs through Cloudflare, Super Speed can work with it in two ways. If you do not use Cloudflare, skip this chapter. Everything is on the **Cloudflare/CDN** tab of the Page cache page.

Level 1: automatic purge (simple)

Switch on **Connect with Cloudflare** and enter your **Zone ID** and an **API Token** that has permission to purge cache. The module checks the details when you save. From then on, when Super Speed clears pages, it also clears them in Cloudflare, so visitors do not see old versions.

- **Cloudflare clear-all purge mode:** "Purge everything" or "Purge by host" (only your domains).
- **Automatically purge Cloudflare cache for updated pages** (On): refreshes the changed URLs when you edit content.

Level 2: full HTML cache (advanced)

Cloudflare itself stores your finished pages and serves them from a location near each visitor. This is the fastest option but also needs the most care, because one saved page is shared by all visitors.

1. Add your domain to Cloudflare and make sure it is **proxied** (orange cloud).
2. In Cloudflare, create a **Cache Rule**: expression **All incoming requests**; Edge TTL **Use cache-control header if present, bypass cache if not**; Browser TTL **Respect origin TTL**.
3. In Super Speed, switch on **Enable Cloudflare/CDN full HTML cache mode**. Set the **Cloudflare HTML cache TTL** (default 24 hours). Leave **Browser cache TTL for Cloudflare HTML pages** at Off unless you know you want it.
4. Switch on **Do not store local HTML cache files in Cloudflare/CDN mode** (recommended). Your server then returns live HTML and Cloudflare keeps the cached copy.
5. Choose the **Default currency** used to build the shared pages. If your shop has several currencies or price groups, switch on **Load product prices dynamically for shared HTML cache**, so each visitor still sees the right price.
6. Test as a guest, as a logged-in customer, and with items in the cart.

> **Important**
> In full HTML mode the shared page is built for one default currency and country, and the separate-by options in 5.3 no longer apply. Personal information is loaded after the page appears. Always test login, cart and checkout carefully before going live.

7. Server cache and minification

Speed Optimization › Cache and minification › Server cache and minification. Five switches, each one making your shop do less work or send less data.

Setting	What it does
Smarty Cache	Keeps PrestaShop's compiled templates so they are not rebuilt on every request. Reduces template rendering time.
Server Cache	Stores query and calculation results (PrestaShop's Smarty caching, filesystem type). Reduces database access time.
Minify HTML	Removes extra spaces, line breaks and comments from page HTML.
Minify CSS	Combines and shrinks your style files into fewer, smaller files.
Minify Javascript	Combines and shrinks your script files into fewer, smaller files.

> **If something looks broken after turning on minify**
> Turn off **Minify Javascript** first, then **Minify CSS**, then **Minify HTML**, testing after each change, until you find the one that conflicts with your theme or a module. Keep the others on. Page cache is cleared automatically when you change these settings.

If your hosting supports **Redis, Memcached or APCu**, enabling one in **Advanced Parameters › Performance** (PrestaShop's own settings) gives a further boost. The Dashboard checklist flags this when it is missing.

8. Browser cache and compression

Speed Optimization › Cache and minification › Browser cache and compression. When this is on, returning visitors reuse files already in their browser instead of downloading them again, and files that travel to the visitor are compressed (Brotli when your server supports it, otherwise Gzip).

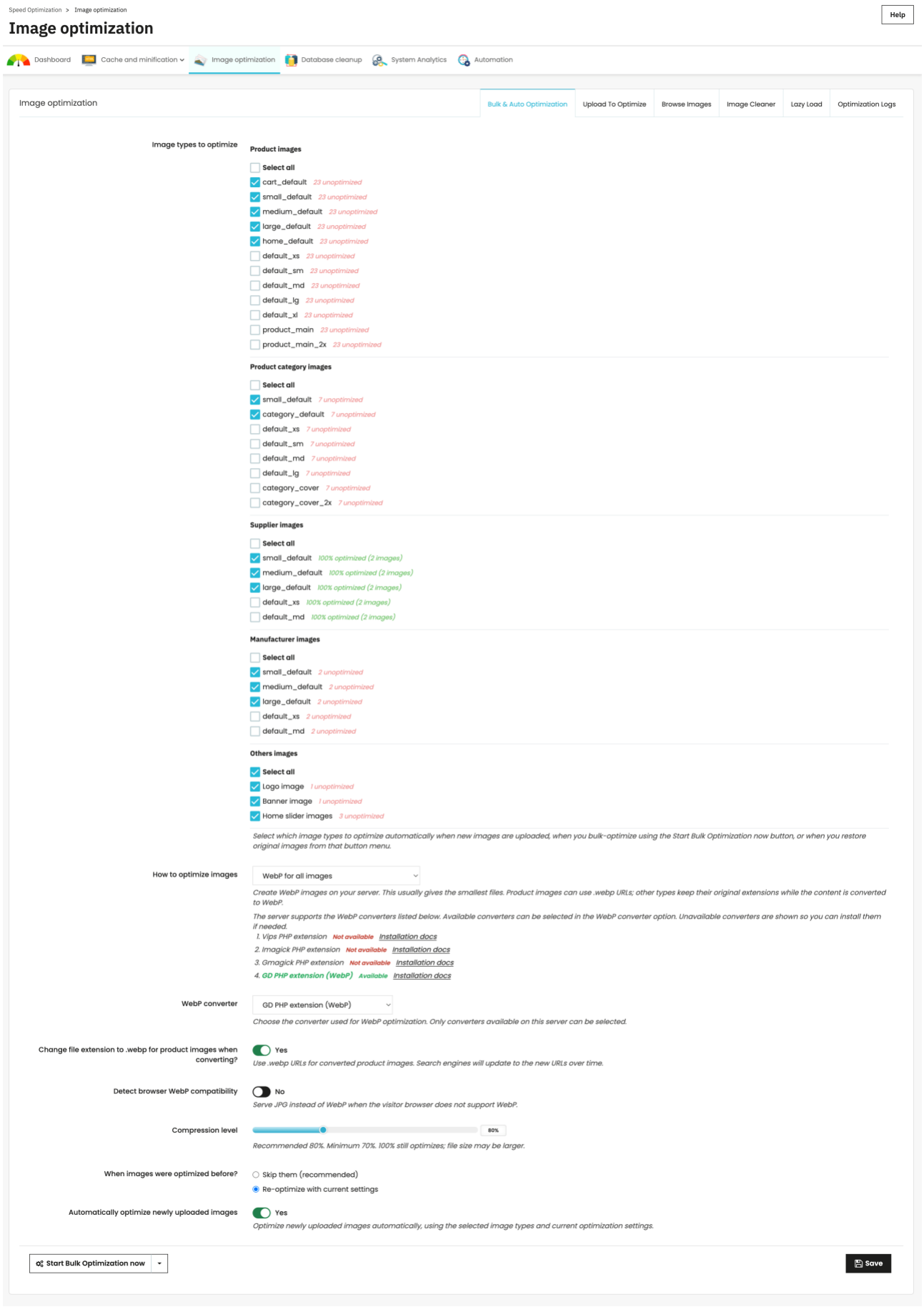
Setting	Default	Notes
Enable browser cache and compression	On	The master switch. Needs Apache modules (the page shows which are available).
Image cache lifetime	12 months	Range 1–36 months.
Icon and SVG cache lifetime	1 year	Range 1–10 years.
CSS cache lifetime	53 weeks	Range 1–160 weeks.
JavaScript cache lifetime	53 weeks	Range 1–160 weeks.
Font cache lifetime	1 year	Range 1–10 years.

The defaults follow Google Lighthouse recommendations, so most shops should not change them. There is a button to reset to the recommended values, and a viewer to read your current `.htaccess` file. The Dashboard checklist marks the feature as good when it is on.

> **Nginx or other servers**
> This feature writes rules into `.htaccess`, which only Apache reads. On Nginx the settings have no effect. Ask your host to enable cache headers and Brotli/Gzip compression in the server configuration.

9. Image optimization

Speed Optimization › Image optimization. Images are usually the heaviest part of a page, so this is where many shops save the most data. The page has six tabs: **Bulk & Auto Optimization**, **Upload to optimize**, **Browse images**, **Image cleaner**, **Lazy load** and **Optimization logs**.


Speed Optimization > Image optimization
Image optimization
Help
Dashboard
Cache and minification
Image optimization
Database cleanup
System Analytics
Automation
Image optimization
Bulk & Auto Optimization
Upload To Optimize
Browse Images
Image Cleaner
Lazy Load
Optimization Logs
Image types to optimize
Product images
Select all
cart_default 23 unoptimized
small_default 23 unoptimized
medium_default 23 unoptimized
large_default 23 unoptimized
home_default 23 unoptimized
default_xs 23 unoptimized
default_sm 23 unoptimized
default_md 23 unoptimized
default_lg 23 unoptimized
default_xl 23 unoptimized
product_main 23 unoptimized
product_main_2x 23 unoptimized
Product category images
Select all
small_default 7 unoptimized
category_default 7 unoptimized
default_xs 7 unoptimized
default_sm 7 unoptimized
default_md 7 unoptimized
default_lg 7 unoptimized
category_cover 7 unoptimized
category_cover_2x 7 unoptimized
Supplier images
Select all
small_default 100% optimized (2 images)
medium_default 100% optimized (2 images)
large_default 100% optimized (2 images)
default_xs 100% optimized (2 images)
default_md 100% optimized (2 images)
Manufacturer images
Select all
small_default 2 unoptimized
medium_default 2 unoptimized
large_default 2 unoptimized
default_xs 2 unoptimized
default_md 2 unoptimized
Others images
Select all
Logo image 1 unoptimized
Banner image 1 unoptimized
Home slider images 3 unoptimized
Select which image types to optimize automatically when new images are uploaded, when you bulk-optimize using the Start Bulk Optimization now button, or when you restore original images from that button menu.
How to optimize images
WebP for all images
Create WebP images on your server. This usually gives the smallest files. Product images can use .webp URLs; other types keep their original extensions while the content is converted to WebP.
The server supports the WebP converters listed below. Available converters can be selected in the WebP converter option. Unavailable converters are shown so you can install them if needed.
1. Vips PHP extension Not available Installation docs
2. Imagick PHP extension Not available Installation docs
3. Gmagick PHP extension Not available Installation docs
4. GD PHP extension (WebP) Available Installation docs
WebP converter
GD PHP extension (WebP)
Choose the converter used for WebP optimization. Only converters available on this server can be selected.
Change file extension to .webp for product images when converting?
Yes
Use .webp URLs for converted product images. Search engines will update to the new URLs over time.
Detect browser WebP compatibility
No
Serve JPG instead of WebP when the visitor browser does not support WebP.
Compression level
80%
Recommended 80%. Minimum 70%. 100% still optimizes; file size may be larger.
When images were optimized before?
Skip them (recommended)
Re-optimize with current settings
Automatically optimize newly uploaded images
Yes
Optimize newly uploaded images automatically, using the selected image types and current optimization settings.
Start Bulk Optimization now
Save

9.1 Choose how to optimize

Method	What to know
Standard compression (recommended)	Done on your own server. No outside services. Default quality 85% (minimum 70%).
WebP for product images only	Converts product photos to WebP, which is usually smaller. Other images use the standard method. Default quality 80%.
WebP for all images	Converts all optimized images to WebP.
reSmush.it compression	Sends images to the free reSmush.it service. Default quality 92% (minimum 90%, JPEG files).
Tinify (TinyPNG/TinyJPG)	Sends images to the Tinify service. Needs an API key from Tinify; you can add several keys, and the module rotates between them. Usage limits are set by Tinify.

WebP conversion needs software on your server (the page shows what is available: vips, Imagick, Gmagick or GD). If none is available, choose a different method.

> **Which should I pick?**
> If unsure, start with **Standard compression**. If your shop has many product photos and your server supports it, **WebP for product images only** is a good step up. Keep the quality at the recommended level: going lower makes pictures look worse for little extra saving.

9.2 Bulk and automatic optimization

- Choose which image types to optimize for each group: products, categories, suppliers, manufacturers, home slider, blog images and others.
- Click **Optimize all images** to process your existing images. You can pause or stop and continue later.
- **When images were optimized before?** “Skip them” (recommended) or “Re-optimize with current settings”.
- **Automatically optimize newly uploaded images** (On): every new image is optimized as soon as you upload it.
- **Restore original images:** available when the chosen method keeps the original files. Use it before uninstalling if you want the originals back.
- **Change file extension to .webp for product images** (Off) and **Detect browser WebP compatibility** (Off). If you turn on the first, search engines gradually learn the new image addresses. The second sends a JPG to the few browsers that cannot show WebP.

9.3 Upload to optimize and Browse images

Upload to optimize lets you drop in images from your computer, optimize them and download the result. **Browse images** lets you pick folders on your server (for example, a theme or module folder)

and optimize the pictures inside. Both keep a history, and you can download or restore individual images.

9.4 Image cleaner

Over time your server collects images that nothing uses any more: pictures of deleted products, old image sizes, and so on. The **Image cleaner** tab finds them in groups (category, supplier, manufacturer, product and multilingual product images) and shows how much space they take. You can delete selected files or all of them.

> **Back up before you delete**
> Deleted images cannot be recovered from the module. Make a backup of the img folder first, especially the first time you use the cleaner.

9.5 Lazy load

With lazy load, images further down the page are loaded only when the visitor gets near them. The top of the page appears sooner, and visitors who do not scroll never download the rest.

- **Enable lazy load** (Off by default).
- **Enable Lazy Load for:** Product listing, Home slider, Home banner. Defaults are all three once lazy load is on.
- **Preloading image:** the animation shown while an image loads: Wave (default), Circle, Ring, Orbit, Dots, Spokes, or None. You can also set the **placeholder background** and **spinner colors** to match your design.

> **It edits a few theme files**
> To make lazy load work, the module edits the image templates of your theme (product miniatures, and the Home slider and Banner modules). It keeps a backup of each original and puts them back when you turn lazy load off or uninstall the module. After you change or update your theme, check that product images still appear; if not, open the Lazy load tab and save again.

10. Database cleanup

Speed Optimization › Database cleanup. A busy shop collects a lot of old data in its database: visitor logs, abandoned carts, mail history and more. This makes the database larger and slower. This page shows how many removable records you have for each item, with a status of Good, Medium or Many, and lets you delete them in one click.

Item	Keep last	Auto (cron)	What it removes
Connections log	30 days	Yes	Old visitor connection records, with their page views.

Item	Keep last	Auto (cron)	What it removes
Orphan page views	30 days	Yes	Page-view records no longer linked to a connection.
Useless discount codes	0 days	No	Expired cart rules and expired specific prices.
Abandoned carts	7 days	No	Carts that never became an order.
Guest data	30 days	No	Visitors without an account or order.
PrestaShop logs	90 days	Yes	Entries in PrestaShop's log table.
Mail history	90 days	Yes	Record of sent emails.
Search statistics	90 days	Yes	Stored search keywords (does not affect your search index).
Module history	180 days	No	Back Office module history.
Super Speed cache logs	30 days	Yes	The module's own skip and error logs.

- You can change the "keep last" period of each item and choose which ones are included in the automatic cleanup.
- Before deleting, you can **download the data as a CSV file**.
- The automatic cleanup is Off until you switch it on in **Automation** (Chapter 12).

> **Deleted data is gone for good**
> Make a database backup before the first cleanup. If you send abandoned-cart reminder emails, keep the **Abandoned carts** period long enough for your reminder flow to finish.

11. System Analytics

Speed Optimization › System Analytics. This page answers "what is still slowing my shop down?" It has two parts.

Checklist

A list of checkpoints with a status of **Good**, **Acceptable**, **Recommended** or **Bad**, and advice for each. It covers the speed features in this module, plus things that often hurt speed: too many products on the home page (the guide values are about 12 per block and 48 per category page), too many images, CSS or JavaScript files on the home page, and a home slider with too many slides. Each line links straight to the page where you can fix it.

Module performance

Every module on your shop adds its own content to the pages through “hooks”, and a slow module slows every page. Switch on **Record module performance** and browse your shop for a while. Then open the **Module performance** tab to see which modules and hooks took longest, filtered by module, hook, page, time or date. Anything slower than about 300 ms deserves a look.

- You can **unhook** a module from a hook directly on this page, and re-hook it later. This removes that module’s output from the pages that use the hook.
- Hooks that could break payment, orders or login are **protected** and cannot be unhooked here.

> **Use recording only while investigating**
> Recording adds a little work to every page. Switch it off when you have finished, then clear the recorded data with the delete button.

12. Automation (cron job)

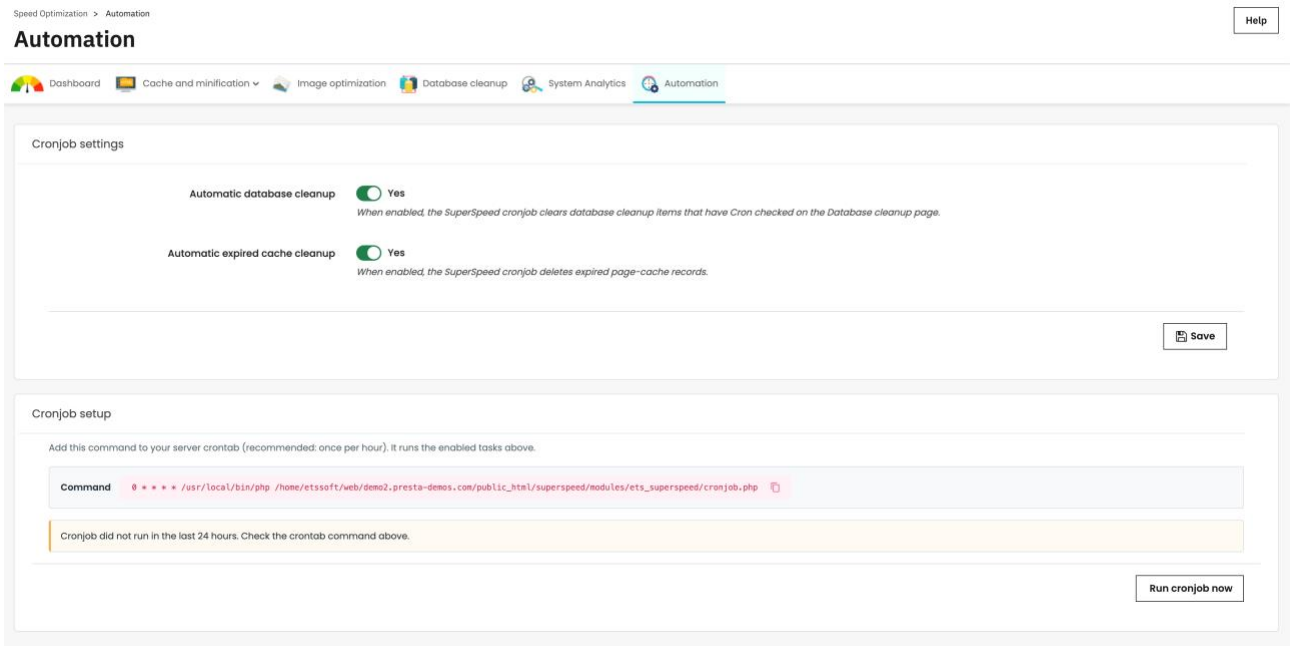
Speed Optimization › Automation. A cron job is a scheduled task on your server. Super Speed uses one to do its housekeeping without you. It does two things, each with its own switch:

Setting	Default	What it does
Automatic expired cache cleanup	On	Deletes saved page copies that have passed their lifetime.
Automatic database cleanup	Off	Runs the items ticked as “Cron” on the Database cleanup page.

Setting it up

1. On the Automation page, copy the **Command**. It looks like this (your path will differ): `0 * * * * /usr/bin/php /path/to/your/shop/modules/ets_superspeed/cronjob.php`
2. In your hosting control panel (for example, cPanel › **Cron Jobs**), add a new cron job and paste the command. **Once per hour** is recommended.
3. Back on the Automation page, click **Run cronjob now** to test. The **Last run** line shows when it last ran and whether it was manual or automatic.

If the page says “Cronjob did not run in the last 24 hours”, check the command and PHP path with your hosting provider. They can set it up for you in a few minutes.


Speed Optimization > Automation
Automation
Help
Dashboard
Cache and minification
Image optimization
Database cleanup
System Analytics
Automation
Cronjob settings
Automatic database cleanup
Yes
When enabled, the SuperSpeed cronjob clears database cleanup items that have Cron checked on the Database cleanup page.
Automatic expired cache cleanup
Yes
When enabled, the SuperSpeed cronjob deletes expired page-cache records.
Save
Cronjob setup
Add this command to your server crontab (recommended: once per hour). It runs the enabled tasks above.
Command
0 * * * * /usr/local/bin/php /home/etssoft/web/demo2.presta-demos.com/public_html/superspeed/modules/ets_superspeed/cronjob.php
Cronjob did not run in the last 24 hours. Check the crontab command above.
Run cronjob now

PART 3

Best settings for your website

13. Choosing your recipe

Every shop is a little different. The recipes below show the settings that matter for each kind of website. **All recipes start from the Quick start (Chapter 3):** Auto configuration, image optimization and the cron job. Then apply the changes listed for your type.

If this describes you…	Use
I show products but do not sell online (no cart or checkout)	Recipe A: Catalogue / showcase site
I sell online in one currency to ordinary customers	Recipe B: Standard online store
I sell in several currencies, languages or countries	Recipe C: Multi-currency / multi-country store
Customers get different prices (wholesale, members, VIP)	Recipe D: Group pricing / B2B store
I have lots of traffic and use Cloudflare	Recipe E: High-traffic store with Cloudflare
My site is mostly articles or a blog	Recipe F: Content / blog-driven site
My hosting is basic, shared or low on disk space	Recipe G: Limited hosting resources

You can combine recipes. For example, a B2B shop on weak hosting uses D and G.

Recipe A: Catalogue / showcase site

Best for: Websites that present products or services but do not take orders online. Visitors rarely log in, and the cart is not used.

Setting	Recommended
Page cache	**On**, all page types ticked
Lifetimes	Home 7 days, Category 7 days, Product 30 days, CMS 30 days (pages refresh automatically when you edit them)
Separate by group / login / cart / country	**Off** (every visitor gets the same page)
Delete product cache after cart and order changes	**Off**
Cloudflare full HTML mode	**Off** (optional: use Level 1 purge only)

Setting	Recommended
Minify HTML, CSS, JavaScript	**On**
Images	WebP for product images, or Standard compression. Lazy load **On** for all three areas.
Database cleanup (Automation)	**On**

Avoid: turning on options that create extra copies, such as login or cart separation. They are not needed and only lower the number of cache hits.

Check afterwards: contact forms still send (forms are never cached), and a recently edited product shows its changes right away.

Recipe B: Standard online store

Best for: A normal shop with one currency, selling to customers who browse as guests or with an account. This is what the default settings are made for.

Setting	Recommended
Page cache and lifetimes	**On** with the **default lifetimes** (Home 5, Category 5, Product 15, Search 1 day)
Separate by group / login / cart / country	**Off** (the cart, sign-in link and customer name are loaded live)
Delete product cache after cart and order changes	**Off** (switch on only if you show live stock numbers)
Delete best-seller cache on new order	**Off** (the 7-day lifetime is enough)
URLs not cached	Keep the default list (cart, order, account, addresses, history…)
Minify	**On** (test checkout after enabling)
Images	WebP for product images. Lazy load **On** for product listings, home slider and banner.
Cron job	Hourly, with expired cache cleanup and database cleanup On

Avoid: caching the cart, checkout and account pages. They are excluded by default, so do not remove those entries from **URLs not cached**.

Check afterwards: add to cart, login, checkout and an order from start to finish. The cart box must update correctly.

Recipe C: Multi-currency / multi-country store

Best for: Shops that show different prices, taxes or messages depending on currency, language or the visitor's location.

Setting	Recommended
Language and currency	Handled automatically (separate copies are always made for each)
Separate by country	**On** if prices include different taxes, or shipping and legal messages differ by country
Separate by state	**Off** (only for state-level taxes such as US sales tax)
Separate by group / login / cart	**Off** unless prices also depend on group
Lifetimes	Defaults. Each language/currency/country combination is cached separately, so the first visitors to each combination wait a little longer.
With Cloudflare full HTML mode	Switch on **Load product prices dynamically** (see Recipe E)
Images	Standard or WebP, lazy load On

Avoid: switching on country **and** state together without a real need: the number of copies multiplies quickly.

Check afterwards: switch currency and country in the shop and confirm prices, taxes and shipping texts are correct on cached pages. Purge all page caches after changing these options.

Recipe D: Group pricing / B2B store

Best for: Wholesale, trade or members-only shops where prices, discounts or even available products depend on the customer's group or login.

Setting	Recommended
Separate by customer group	**On**
Separate by all customer groups	**On** only if customers belong to several groups that affect prices; otherwise Off
Separate by login status	**On** if logged-in customers see different content than guests (for example hidden prices)
Separate by cart status	**Off** (cart contents load live)
Lifetimes	Defaults; keep Search at 1 day
URLs not cached	Keep defaults; add any custom portal or quote pages
Images	Standard or WebP, lazy load On

Avoid: assuming prices are right without testing. Cached pages must match the customer's group.

Check afterwards: log in with a test customer from each group, and as a guest, and check that each sees the correct prices. Then log out and confirm that no price from the previous login is visible.

Recipe E: High-traffic store with Cloudflare

Best for: Shops with heavy traffic, or visitors from many countries, that already use Cloudflare. The goal is to serve most pages from Cloudflare's servers around the world.

Setting	Recommended
Connect with Cloudflare	**On** with Zone ID and API Token (cache purge permission)
Enable Cloudflare/CDN full HTML cache mode	**On**, with the Cache Rule from Chapter 6
Cloudflare HTML cache TTL	Default 24 hours
Do not store local HTML cache files	**On**
Default currency for Cloudflare HTML pages	Your main currency
Load product prices dynamically	**On** if you have several currencies or price groups; Off if prices are the same for everyone
Automatically purge Cloudflare cache for updated pages	**On**
Cron job	Hourly

Avoid: enabling full HTML mode before the Cloudflare Cache Rule is in place, or without testing logged-in and cart pages.

Check afterwards: test as a guest, as a logged-in customer and with a cart, from a second device. Edit a product and confirm that the change appears on the front end.

Recipe F: Content / blog-driven site

Best for: Sites where articles, guides or news are the main attraction, often alongside a small shop. Works with the PrestaHero Blog module.

Setting	Recommended
Content to cache	Tick **Blog pages** (appears when the blog module is installed), plus Home, CMS and Category
Lifetimes	Blog 7 days (default); longer for old archives is fine. Pages refresh when you edit a post.

Setting	Recommended
Query parameters	Keep the default. Marketing links with tracking codes will not create extra copies.
Images	Optimize **Blog post, Blog category, Blog gallery and slider** image types in the Image optimization tab
Lazy load	**On** (long articles with many images benefit most)
Minify	**On**

Avoid: forgetting to tick the blog image types. They are not optimized unless you select them.

Check afterwards: publish a new post and confirm it appears on the blog page and the home page straight away.

Recipe G: Limited hosting resources

Best for: Shared hosting, small plans, or a large catalogue on a small disk. The aim is to get speed without overloading the server.

Setting	Recommended
Page cache	**On** (the single biggest gain for weak servers)
Separate-by options	**Off** unless you really need them. Fewer copies mean less disk and CPU.
Search page lifetime	1 day (keep short)
Compress cache file	**Off** normally; On only if disk space is the real problem
Image method	Standard compression. Run **bulk optimization** at night and use Pause if the server struggles.
Image cleaner	Run once to free space
Tracking and logs (hit counters, skip logs, record module performance)	**Off**
Database cleanup	Automatic cleanup On, with shorter “keep” periods for logs
Redis / Memcached / APCu	Enable in Advanced Parameters › Performance if your host offers it

Avoid: optimizing all images at once during busy hours, and leaving debug tracking switched on.

Check afterwards: the Dashboard has no warnings, and the “Page cache” and “Cron” items are healthy.

14. Comparison table

A one-page summary of the recipes. “n/a” means the option does not apply in that mode.

Setting	A Catalogue	B Standard	C Multi-region	D Group pricing	E Cloudflare	F Blog	G Low resource
Page cache	On	On	On	On	On	On	On
Separate by customer group	Off	Off	Off	On	n/a	Off	Off
Separate by login status	Off	Off	Off	If needed	n/a	Off	Off
Separate by cart status	Off	Off	Off	Off	n/a	Off	Off
Separate by country	Off	Off	If needed	Off	n/a	Off	Off
Cloudflare full HTML mode	Off	Off	Off	Off	On	Off	Off
Dynamic price loading	Off	Off	Off	Off	If prices vary	Off	Off
Lazy load	On	On	On	On	On	On	On
Product image method	WebP	WebP	Std / WebP	Std / WebP	WebP	WebP	Standard
Minify HTML, CSS, JS	On	On	On	On	On	On	On
Automatic cron cleanup	On	On	On	On	On	On	On
Hit counters and logs	Off	Off	Off	Off	Off	Off	Off

PART 4

Troubleshooting and support

15. After you change settings

1. **Purge all caches** from the Dashboard so you test with fresh pages.
2. **Test the key journeys** in a private browser window: home page, category, product, search, add to cart, login, checkout.
3. **Test on mobile** as well as desktop.
4. **Measure** your speed again. Repeat the test twice (first visit builds the cache).
5. **Keep an eye on it** for a week or two. Switch off any hit counters or logs you turned on for testing.

A little maintenance goes a long way. Every month, glance at the Dashboard checklist and System Analytics, run the Image cleaner if you add many products, and check that the cron job's **Last run** is recent.

16. Troubleshooting

What you see	Likely cause	What to do
I changed a product but the shop still shows the old version	The change was made outside the Back Office (import, ERP, database), so the cache did not know. Cloudflare is still holding the old page.	Click **Purge all caches** on the Dashboard. If you use Cloudflare, connect it so it is purged too.
A visitor sees the wrong price, currency or group price	Prices depend on something the cache does not separate by.	Turn on the matching **Separate page cache by…** option (group, country…) or use dynamic prices with Cloudflare, then purge all caches.
Cart count or "Sign in" link is wrong	A theme module that shows live info is not in the dynamic list.	Add that module and its hook in the **Dynamic content** tab, then purge all caches.
Layout is broken after enabling minify	One of the minify options conflicts with the theme or a module.	Turn off Minify JavaScript, then CSS, then HTML, testing after each. Keep the rest on.

What you see	Likely cause	What to do
A page is never cached	It matches a rule that skips the cache.	Check the X-SS header (Appendix). Turn on **Log skipped cache creation** and read the **Cache skip logs** tab.
Product images are missing or still JPG after switching to WebP	Your server cannot convert to WebP, or the new image addresses are not picked up yet.	Check the WebP converter status on the Image optimization page. Turn off the “change file extension to .webp” option. Use **Restore original images** if needed.
Images are blank or jump around after enabling lazy load	The theme’s image markup is unusual.	Turn lazy load off for the area that misbehaves (product listing, slider or banner), or turn it off entirely, and save again.
The cron job does not run	Wrong PHP path or command; cron not enabled in hosting.	Compare with the command on the Automation page, run **Run cronjob now**, and ask your host to check the PHP path.
Dashboard warns that Cloudflare login failed	The API token or Zone ID is wrong or expired.	Create a new token with cache purge permission and save it again on the Cloudflare/CDN tab.
Dashboard shows an override or install log warning	Another module also changes the same PrestaShop core files.	Note the file names shown on the Dashboard and contact PrestaHero support. Do not delete files from the override folder yourself.
The Back Office feels slower	Recording or hit counters are switched on.	Turn off **Record module performance** and the hit counters and logs.
Pages are slow right after purging	This is normal. Each page is rebuilt on its first visit.	Avoid purging all caches during busy hours. Purge only the affected page when you can.

17. Turning things off

You can step back as far as you need:

1. **One feature:** switch it off on its own page and save. For example, turn off Lazy load or Minify JavaScript.
2. **All caches:** click **Disable all caches** on the Dashboard. Your shop works as before.
3. **The module:** disable it from **Modules**. Cached files are not used while it is disabled.
4. **Uninstall:** if you want the original images back, use **Restore original images** first. On uninstall, the module removes its own tables, settings, cache files and tabs, and puts back the theme files it edited for lazy load.

18. FAQ

Will customers see someone else's cart or account?

No. The cart, account and checkout pages are excluded from the cache by default, form submissions are never cached, and personal parts of a page are loaded separately for each visitor.

How quickly do my edits show on the shop?

Immediately for changes made in the Back Office. The module clears the affected pages as soon as you save.

Do I really need the cron job?

The shop works without it, but expired cache files and old database records then pile up. It takes five minutes to set up, so it is worth doing.

How do I know the cache is working?

Check the `X-SS` response header: HIT means the page came from the cache (see Chapter 5.6). The Cached pages list also shows what is stored.

Does it work with other CDNs?

Full HTML caching is built for Cloudflare. Other CDNs can serve your images and files through PrestaShop's media server settings.

Does it work with several shops (multi-shop)?

Yes. Cache and cleanup work per shop.

Why is my first visit to a page not fast?

The first visit builds the saved copy. Later visits are fast.

Appendix: cache status codes

The `X-SS` header in your browser's developer tools starts with one of these codes. It also shows how long the server took (`spt`, in milliseconds) and, for a HIT, when the saved copy expires (`never` means Forever).

Code	Meaning
HIT	The page came from the cache.
MISS	Not in the cache yet. It was built now and saved if allowed.

Code	Meaning
DYNAMIC	Not cached: a form submission, a background (AJAX) request, or a page type that is not ticked in Content to cache.
DISABLED	Page cache is switched off.
BYPASS_EXCEPTION	The address matches an entry in URLs not cached.
BYPASS_QUERY	The address has parameters the module does not recognise (in the default query parameter mode).
BYPASS_ACTION	The request performs an action (for example an add-to-cart link), so it is not cached.
BYPASS_404 / BYPASS_STATUS	An error page or a response that is not a normal “200 OK” is never saved.
BYPASS_API	The address belongs to the PrestaShop web service (API).
BYPASS_ADMIN	The request comes from a logged-in Back Office user.
BYPASS_SHOP_DISABLED	The shop is in maintenance mode.
BYPASS_IP_LIMIT	One visitor created too many copies in a short time (protection setting).
MISS_METADATA, BYPASS_CDN_METADATA	Seen in Cloudflare full HTML mode, where only a record of the page is kept on your server.