MULTIGURA

Performance benchmark: MULTIGURA vs Polylang

Front-end and admin page load, write operations and throughput on a bare WordPress site, with Elementor, Divi 5 and Avada, and in a WooCommerce shop. Each plugin runs on its own standalone single-site WordPress (no multisite) with identical content, next to a site without a multilingual plugin, all measured side by side in interleaved rounds.

Versions compared

Component	Version
PHP	**8.3.37** (PHP-FPM, 5 workers) + OPcache (`validate_timestamps=0` , 1 GB)
Web server	nginx 1.27.5
Database	MariaDB 11.8.2
WordPress	**7.1.5**, three standalone single-site installs (no multisite)
MULTIGURA	**2.0.16**
Polylang	**Polylang Pro 3.9.2**; Polylang for WooCommerce 2.2.7 (WooCommerce scenario)
WooCommerce	11.3.1 (WooCommerce scenario, on all three sites)
Themes	Twenty Twenty-Five 1.4 · Astra 4.13.1 · Divi 5.19.1 · Avada 7.16.4 · Storefront 4.6.2
Builders	Elementor 4.4.1 + Elementor Pro 4.4.1 · Divi 5 (theme) · Avada Builder 3.16.4 + Avada Core 5.16.4
Machine	Docker Desktop on Apple Silicon (macOS), 8 vCPU, WordPress files bind-mounted

Date
2026-11-24

Requests measured
5,780 (errors: 0)

Scenarios
5: bare, Elementor, Divi 5, Avada, WooCommerce

1. Summary

+5 ms
MULTIGURA, JA page
vs. no plugin (+8%)

+13 ms
Polylang, JA page
vs. no plugin (+22%)

52 / 52
faster than Polylang
page × language × scenario

- **MULTIGURA adds little to front-end pages.** Averaged over all page types and the 5 scenarios, an English page takes **+4 ms** (+7%) more than the same site without a multilingual plugin, a Japanese page **+5 ms** (+8%). It was faster than Polylang in **52 of 52** page, language and scenario combinations.
- **Polylang** adds **+10 ms** (+18%) to English pages and **+13 ms** (+22%) to Japanese pages; on Japanese pages it needs **1.13×** the time of MULTIGURA. Both plugins store a translation as its own post; Polylang loads about 1 372 PHP files per front-end request (MULTIGURA 1 166, WordPress alone 1 124).
- **Translated pages cost MULTIGURA almost nothing extra.** A translation is rendered like the original: Japanese minus English is +0.6 ms on the content pages; in the shop Japanese pages take +0.8 ms more (product translation lookups), the cart +0.4 ms (section 4).
- **WooCommerce shop:** MULTIGURA adds +6 ms (EN) / +7 ms (JA) to shop pages, Polylang with Polylang for WooCommerce +15 / +18 ms (section 4).
- **Under load:** with four concurrent clients MULTIGURA served **30.2** Japanese pages per second on average, Polylang 27.1 (no plugin: 31.6 English pages per second).
- **Admin and writes:** admin screens average 92 ms with MULTIGURA (no plugin 86 ms, Polylang 102 ms); write requests are dominated by WordPress itself (sections 5 and 6).

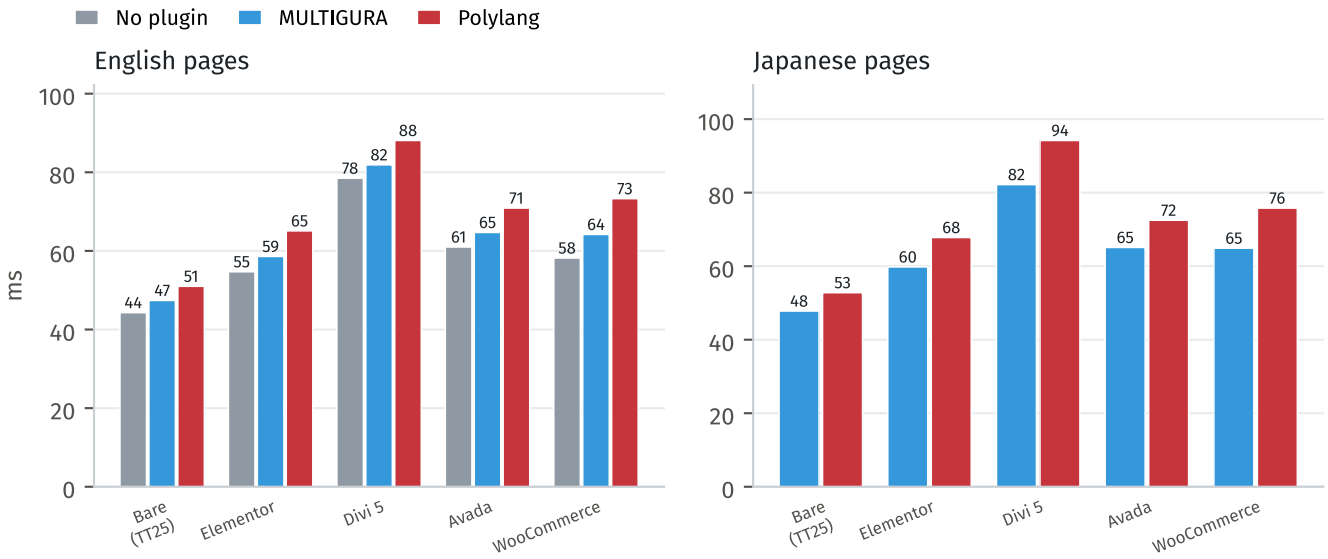

Figure 1. Mean of the median full-response times of the front-end page types per scenario (content scenarios: home, post, page, category, search; WooCommerce: home, shop, product, category, cart, search). Lower is better. Left: English (default language); right: the Japanese translation. “No plugin” has no translation, so it appears only on the left.

2. Test setup and method

Sites

Three **standalone single-site WordPress installs**: **No plugin** (baseline), **MULTIGURA** and **Polylang**. They share the WordPress, plugin and theme files of one installation, but each has its own database tables (table prefix) and uploads folder; nothing is shared between them and none of them is a multisite network. The request header `X-Bench-Site` selects the site in `wp-config.php`, so all three are served by the same nginx and PHP-FPM workers and are measured in the same interleaved rounds. Each translation plugin is active only on its own site.

Content

- Same generator with a fixed random seed on every site: 300 posts (about 450 words, 1 or 2 categories, 3 tags, a featured image), 20 pages (5 with child pages), 12 images, 10 categories, 30 tags, a primary menu with 11 items;
- four landing pages with the same visible content (hero, six feature boxes, a grid of the 6 latest posts, a call to action) built with blocks, Elementor, Divi 5 and Avada Builder;
- WooCommerce: 8 product categories, 60 simple products (price, SKU, stock, image, 1 or 2 categories, short and long description; 6 featured, 12 on sale), the shop, cart, checkout and account pages, and a shop home page on Storefront's homepage template (product categories, recent, featured, popular and on-sale products);
- a Japanese (JA) translation of everything in each plugin's own data model. MULTIGURA: a separate post per language linked in its translations table, category names in term meta, a translated menu. Polylang: a separate post, category and tag per language, with its language and translation group stored as taxonomy terms (created through Polylang's API), and a menu per language in its settings. The Japanese text is a letter-for-letter transliteration of the English into katakana, so translated pages have the same length everywhere.

Two languages: English (default) and Japanese. Permalinks `/%postname%/`. MULTIGURA: posts, pages, categories (and products, product categories in the shop) translatable, media module on, default language at the root (no prefix), as on the other site. Polylang Pro: English and Japanese (`ja`); language in directories, default language without a prefix, the front page of each language at its directory, media translation on.

Scenarios

Bare	Twenty Twenty-Five (block theme), front page built with blocks
Elementor	Astra + Elementor + Elementor Pro; Elementor front page (containers, heading, text, buttons, icon boxes, Recent Posts widget)
Divi 5	Divi theme; Divi 5 front page (heading, text, button, blurbs, Blog module)
Avada	Avada + Avada Builder + Avada Core; Avada front page (title, text, button, columns, Blog element)
WooCommerce	Storefront + WooCommerce on all sites (+ Polylang for WooCommerce on the Polylang site); WooCommerce is off in the other scenarios

What was measured

- **Front-end** (anonymous visitor), English and Japanese: home, single post, child page, category archive, search; in the shop: shop home, shop page, product, product category, cart, product search.
- **Admin** (administrator): Dashboard, Posts and Pages lists, block editor for a post, Categories, Menus (classic themes), the builder's editor; in the shop: Products list, product editor, Product categories, Pages, WooCommerce settings.
- **Writes** over HTTP, as an editor does them: REST create, update and delete of a post, REST create of a category, saving the builder landing page and saving a Japanese translation (creating a JA post). In the shop: WooCommerce REST create, update and delete of a product, a JA product translation, and a guest "add to cart" on the English and the Japanese site (`?wc-ajax=add_to_cart` , new session each round). Plus a bulk test in one WP-CLI process (100 inserts, updates, translations, deletes).
- **Throughput:** 120 requests per URL with 4 concurrent clients (PHP-FPM has 5 workers) on home and post (shop: home and product), two passes in alternating order.

Procedure. Each URL was first requested 3 to 5 times (OPcache, builder CSS). Then 20 rounds; each round requests every URL of every site in a new random order, so drift on the machine affects all sites equally. Requests go from the PHP container to nginx over the Docker network with gzip and keep-alive. A temporary must-use plugin records PHP time from request start to shutdown, database queries, peak memory and included PHP files per request. Values are medians (p90 in the tables). OPcache with `validate_timestamps=0` as on a production server; no page cache, no persistent object cache, WP_DEBUG off, WP-Cron off on the test sites; MULTIGURA's own Memcached cache off. 5,780 measured requests, 0 errors.

Absolute times on Docker Desktop for macOS differ from a Linux server; the report compares the plugins with each other.

3. Front-end pages

Full response time (curl total: request to last byte), median of 20 interleaved rounds. The figure shows the Japanese pages of the multilingual plugins; the tables give both languages with p90, database queries and peak memory.

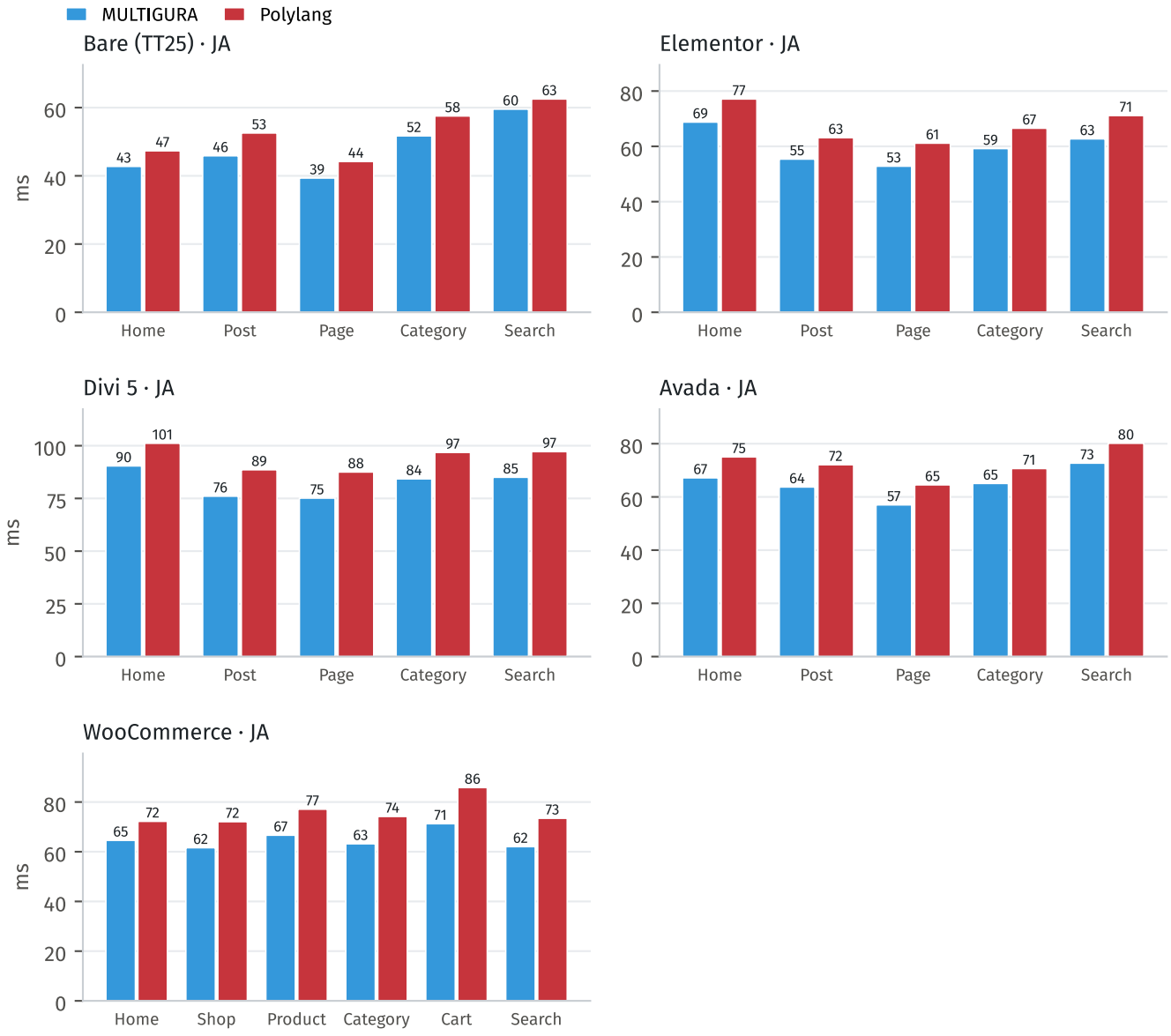

Figure 2. Japanese front-end pages per scenario, median full-response time in ms (lower is better).

Bare (TT25): front-end, ms (median)

Page	Lang	No plugin	MULTIGURA	Polylang
Home	EN	**39** p90 44 · 31 q · 6.2 MB	**42** p90 44 · 39 q · 6.6 MB	**47** p90 50 · 43 q · 7.0 MB
Home	JA	–	**43** p90 47 · 43 q · 6.6 MB	**47** p90 53 · 40 q · 7.4 MB
Post	EN	**43** p90 45 · 38 q · 6.2 MB	**47** p90 54 · 46 q · 6.6 MB	**51** p90 57 · 53 q · 7.1 MB
Post	JA	–	**46** p90 50 · 46 q · 6.6 MB	**53** p90 56 · 58 q · 7.4 MB
Page	EN	**37** p90 41 · 25 q · 6.2 MB	**39** p90 45 · 31 q · 6.6 MB	**43** p90 45 · 40 q · 7.0 MB
Page	JA	–	**39** p90 40 · 35 q · 6.6 MB	**44** p90 48 · 38 q · 7.4 MB
Category	EN	**49** p90 55 · 33 q · 6.3 MB	**51** p90 53 · 40 q · 6.7 MB	**55** p90 59 · 43 q · 7.3 MB
Category	JA	–	**52** p90 58 · 40 q · 6.8 MB	**58** p90 62 · 47 q · 7.7 MB
Search	EN	**53** p90 56 · 34 q · 6.3 MB	**58** p90 61 · 46 q · 6.7 MB	**59** p90 64 · 42 q · 7.2 MB
Search	JA	–	**60** p90 65 · 49 q · 6.8 MB	**63** p90 69 · 47 q · 7.7 MB

Bold: median full-response time; then p90, database queries and PHP peak memory of the request.

Elementor: front-end, ms (median)

Page	Lang	No plugin	MULTIGURA	Polylang
Home	EN	**64** p90 65 · 110 q · 11.5 MB	**67** p90 70 · 119 q · 12.0 MB	**74** p90 79 · 144 q · 12.7 MB
Home	JA	–	**69** p90 74 · 123 q · 12.1 MB	**77** p90 81 · 142 q · 13.5 MB
Post	EN	**50** p90 52 · 100 q · 11.1 MB	**54** p90 59 · 111 q · 11.5 MB	**61** p90 63 · 137 q · 12.1 MB
Post	JA	–	**55** p90 59 · 112 q · 11.5 MB	**63** p90 67 · 138 q · 12.6 MB
Page	EN	**49** p90 50 · 93 q · 11.1 MB	**52** p90 57 · 103 q · 11.5 MB	**60** p90 62 · 130 q · 12.0 MB
Page	JA	–	**53** p90 57 · 106 q · 11.5 MB	**61** p90 64 · 130 q · 12.6 MB
Category	EN	**53** p90 58 · 103 q · 11.2 MB	**58** p90 62 · 114 q · 11.7 MB	**64** p90 69 · 131 q · 12.3 MB
Category	JA	–	**59** p90 62 · 115 q · 11.7 MB	**67** p90 70 · 131 q · 12.9 MB
Search	EN	**57** p90 60 · 98 q · 11.2 MB	**61** p90 63 · 109 q · 11.6 MB	**67** p90 70 · 119 q · 12.2 MB
Search	JA	–	**63** p90 69 · 111 q · 11.7 MB	**71** p90 78 · 121 q · 12.9 MB

Bold: median full-response time; then p90, database queries and PHP peak memory of the request.

Divi 5: front-end, ms (median)

Page	Lang	No plugin	MULTIGURA	Polylang
Home	EN	**86** p90 88 · 75 q · 13.2 MB	**90** p90 92 · 83 q · 13.6 MB	**96** p90 98 · 106 q · 14.2 MB
Home	JA	–	**90** p90 92 · 87 q · 13.7 MB	**101** p90 102 · 102 q · 16.3 MB
Post	EN	**73** p90 74 · 63 q · 13.2 MB	**77** p90 84 · 71 q · 13.6 MB	**83** p90 85 · 97 q · 14.1 MB
Post	JA	–	**76** p90 78 · 72 q · 13.6 MB	**89** p90 92 · 98 q · 16.2 MB
Page	EN	**72** p90 74 · 61 q · 13.2 MB	**75** p90 77 · 70 q · 13.6 MB	**82** p90 83 · 95 q · 14.1 MB
Page	JA	–	**75** p90 77 · 73 q · 13.6 MB	**88** p90 89 · 99 q · 16.2 MB
Category	EN	**80** p90 82 · 86 q · 13.3 MB	**84** p90 85 · 95 q · 13.7 MB	**90** p90 91 · 111 q · 14.2 MB
Category	JA	–	**84** p90 85 · 96 q · 13.7 MB	**97** p90 99 · 117 q · 16.3 MB
Search	EN	**81** p90 84 · 73 q · 13.3 MB	**83** p90 84 · 82 q · 13.7 MB	**89** p90 92 · 98 q · 14.2 MB
Search	JA	–	**85** p90 98 · 84 q · 13.7 MB	**97** p90 103 · 99 q · 16.3 MB

Bold: median full-response time; then p90, database queries and PHP peak memory of the request.

Avada: front-end, ms (median)

Page	Lang	No plugin	MULTIGURA	Polylang
Home	EN	**63** p90 64 · 61 q · 9.0 MB	**67** p90 69 · 69 q · 9.4 MB	**73** p90 74 · 91 q · 10.1 MB
Home	JA	–	**67** p90 69 · 73 q · 9.5 MB	**75** p90 78 · 93 q · 10.6 MB
Post	EN	**59** p90 60 · 68 q · 8.8 MB	**63** p90 65 · 78 q · 9.2 MB	**71** p90 74 · 110 q · 10.0 MB
Post	JA	–	**64** p90 65 · 79 q · 9.3 MB	**72** p90 74 · 111 q · 10.4 MB
Page	EN	**54** p90 54 · 46 q · 8.7 MB	**57** p90 59 · 54 q · 9.1 MB	**63** p90 65 · 81 q · 9.8 MB
Page	JA	–	**57** p90 58 · 57 q · 9.1 MB	**65** p90 65 · 81 q · 10.2 MB
Category	EN	**60** p90 63 · 59 q · 8.9 MB	**64** p90 66 · 70 q · 9.3 MB	**69** p90 71 · 85 q · 10.1 MB
Category	JA	–	**65** p90 68 · 71 q · 9.4 MB	**71** p90 72 · 86 q · 10.6 MB
Search	EN	**68** p90 69 · 59 q · 8.9 MB	**72** p90 74 · 70 q · 9.4 MB	**78** p90 79 · 81 q · 10.2 MB
Search	JA	–	**73** p90 76 · 72 q · 9.6 MB	**80** p90 81 · 84 q · 10.7 MB

Bold: median full-response time; then p90, database queries and PHP peak memory of the request.

4. WooCommerce shop

The shop scenario runs Storefront with WooCommerce on every site; the Polylang site also runs Polylang for WooCommerce, as a Polylang shop would. Mean over the six shop pages, compared with the shop without a multilingual plugin: MULTIGURA +6 ms EN / +7 ms JA, Polylang +15 / +18 ms. The cart is the heaviest page for every site: WooCommerce renders it as a block page with its settings and a product list inlined.

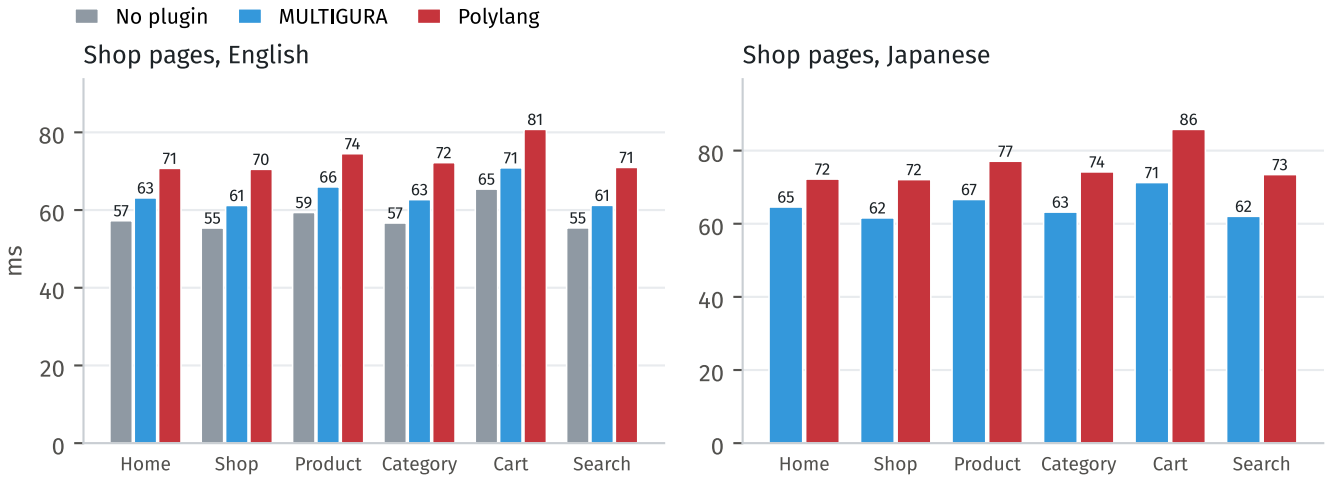

Figure 3. Shop pages, median full-response time in ms (lower is better). Left: English; right: Japanese.

WooCommerce: front-end, ms (median)

Page	Lang	No plugin	MULTIGURA	Polylang
Home	EN	**57** p90 59 · 108 q · 11.1 MB	**63** p90 68 · 121 q · 11.7 MB	**71** p90 73 · 154 q · 12.4 MB
Home	JA	–	**65** p90 66 · 131 q · 11.9 MB	**72** p90 76 · 151 q · 13.4 MB
Shop	EN	**55** p90 57 · 80 q · 11.1 MB	**61** p90 63 · 90 q · 11.8 MB	**70** p90 78 · 125 q · 12.4 MB
Shop	JA	–	**62** p90 66 · 94 q · 11.9 MB	**72** p90 75 · 123 q · 13.6 MB
Product	EN	**59** p90 61 · 99 q · 11.1 MB	**66** p90 67 · 118 q · 11.6 MB	**74** p90 77 · 152 q · 12.4 MB
Product	JA	–	**67** p90 70 · 120 q · 11.7 MB	**77** p90 78 · 154 q · 13.5 MB
Category	EN	**57** p90 58 · 84 q · 11.2 MB	**63** p90 64 · 96 q · 11.8 MB	**72** p90 73 · 134 q · 12.5 MB
Category	JA	–	**63** p90 65 · 100 q · 11.9 MB	**74** p90 77 · 132 q · 13.7 MB
Cart	EN	**65** p90 69 · 98 q · 12.1 MB	**71** p90 73 · 108 q · 12.7 MB	**81** p90 82 · 146 q · 13.4 MB
Cart	JA	–	**71** p90 78 · 115 q · 12.8 MB	**86** p90 88 · 144 q · 15.5 MB
Search	EN	**55** p90 58 · 79 q · 11.2 MB	**61** p90 65 · 89 q · 11.7 MB	**71** p90 73 · 123 q · 12.5 MB
Search	JA	–	**62** p90 65 · 94 q · 12.0 MB	**73** p90 75 · 126 q · 13.7 MB

Bold: median full-response time; then p90, database queries and PHP peak memory of the request.

Shop writes

Server time (PHP), median of 15 rounds. Product create, update and delete (WooCommerce REST API, mean): no plugin 45 ms, MULTIGURA 48 ms, Polylang 54 ms. Guest add to cart on the Japanese site: MULTIGURA 31 ms, Polylang 38 ms (no plugin, English: 25 ms).

Operation	No plugin	MULTIGURA	Polylang
Create product	**49** 253 q	**54** 276 q	**60** 290 q
Update product	**39** 125 q	**42** 142 q	**45** 133 q
Save JA product translation	–	**52** 256 q	**63** 311 q
Add to cart (EN, guest)	**25** 51 q	**27** 57 q	**31** 72 q
Add to cart (JA, guest)	–	**31** 62 q	**38** 80 q
Delete product	**46** 221 q	**50** 249 q	**58** 259 q

Shop admin screens (PHP time, ms)

Screen	No plugin	MULTIGURA	Polylang
Dashboard	**76** 118 q	**79** 118 q	**89** 137 q
Pages list	**84** 117 q	**92** 120 q	**104** 131 q
Products list	**118** 136 q	**131** 138 q	**145** 199 q
Edit product	**107** 139 q	**112** 154 q	**122** 171 q
Product categories	**97** 116 q	**100** 114 q	**117** 135 q
WooCommerce settings	**91** 111 q	**94** 108 q	**102** 127 q

5. Admin screens

Server time (PHP) of admin screens, mean over the screens of all scenarios: no plugin 86 ms, MULTIGURA 92 ms, Polylang 102 ms. Builder editors are dominated by the builder itself. Polylang loads about 1 394 PHP files per admin request (MULTIGURA 1 149, WordPress alone 1 105). The WooCommerce screens are in section 4.

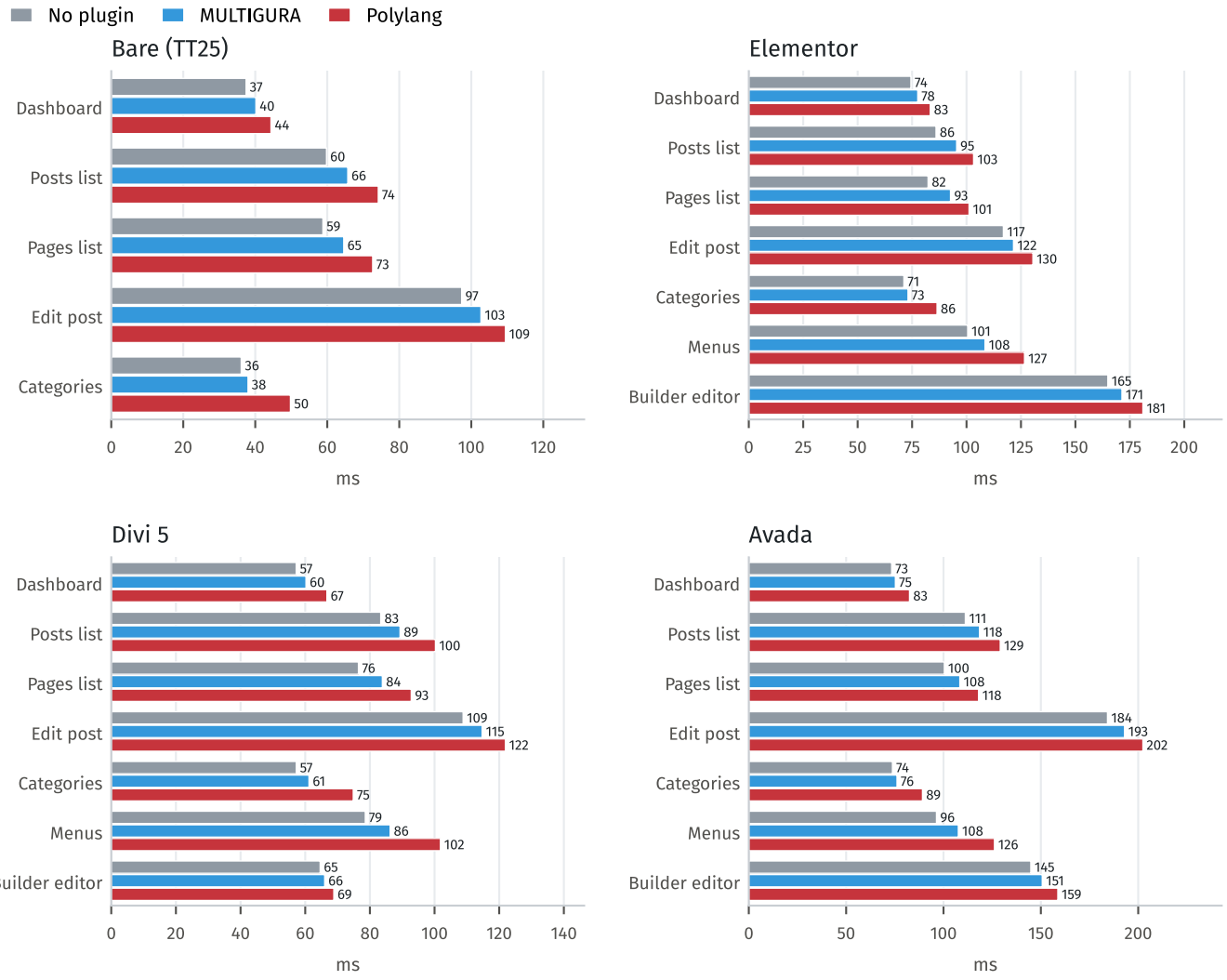

Figure 4. Admin screens, server (PHP) time in ms, median (lower is better).

Scenario	Screen	No plugin	MULTIGURA	Polylang
Bare (TT25)	Dashboard	**37** 39 q	**40** 43 q	**44** 47 q
Bare (TT25)	Posts list	**60** 44 q	**66** 48 q	**74** 56 q
Bare (TT25)	Pages list	**59** 36 q	**65** 43 q	**73** 45 q
Bare (TT25)	Edit post	**97** 56 q	**103** 76 q	**109** 73 q
Bare (TT25)	Categories	**36** 29 q	**38** 31 q	**50** 38 q
Elementor	Dashboard	**74** 116 q	**78** 122 q	**83** 128 q
Elementor	Posts list	**86** 115 q	**95** 141 q	**103** 129 q
Elementor	Pages list	**82** 107 q	**93** 136 q	**101** 138 q
Elementor	Edit post	**117** 138 q	**122** 159 q	**130** 156 q
Elementor	Categories	**71** 100 q	**73** 104 q	**86** 110 q

Scenario	Screen	No plugin	MULTIGURA	Polylang
Elementor	Menus	**101** 217 q	**108** 227 q	**127** 294 q
Elementor	Builder editor	**165** 140 q	**171** 147 q	**181** 151 q
Divi 5	Dashboard	**57** 55 q	**60** 61 q	**67** 67 q
Divi 5	Posts list	**83** 59 q	**89** 63 q	**100** 73 q
Divi 5	Pages list	**76** 51 q	**84** 57 q	**93** 63 q
Divi 5	Edit post	**109** 83 q	**115** 103 q	**122** 100 q
Divi 5	Categories	**57** 44 q	**61** 48 q	**75** 55 q
Divi 5	Menus	**79** 157 q	**86** 163 q	**102** 206 q
Divi 5	Builder editor	**65** 50 q	**66** 52 q	**69** 55 q
Avada	Dashboard	**73** 71 q	**75** 75 q	**83** 80 q
Avada	Posts list	**111** 98 q	**118** 103 q	**129** 107 q
Avada	Pages list	**100** 66 q	**108** 74 q	**118** 76 q
Avada	Edit post	**184** 131 q	**193** 151 q	**202** 157 q
Avada	Categories	**74** 63 q	**76** 65 q	**89** 73 q
Avada	Menus	**96** 210 q	**108** 254 q	**126** 327 q
Avada	Builder editor	**145** 94 q	**151** 98 q	**159** 112 q

6. Write operations

Server time (PHP) of write requests, median of 15 rounds. Mean over create, update and delete post, create category and builder save: no plugin 92 ms, MULTIGURA 94 ms, Polylang 106 ms. Most of the time is WordPress core (revisions, term counts, caches). The WooCommerce writes are in section 4.

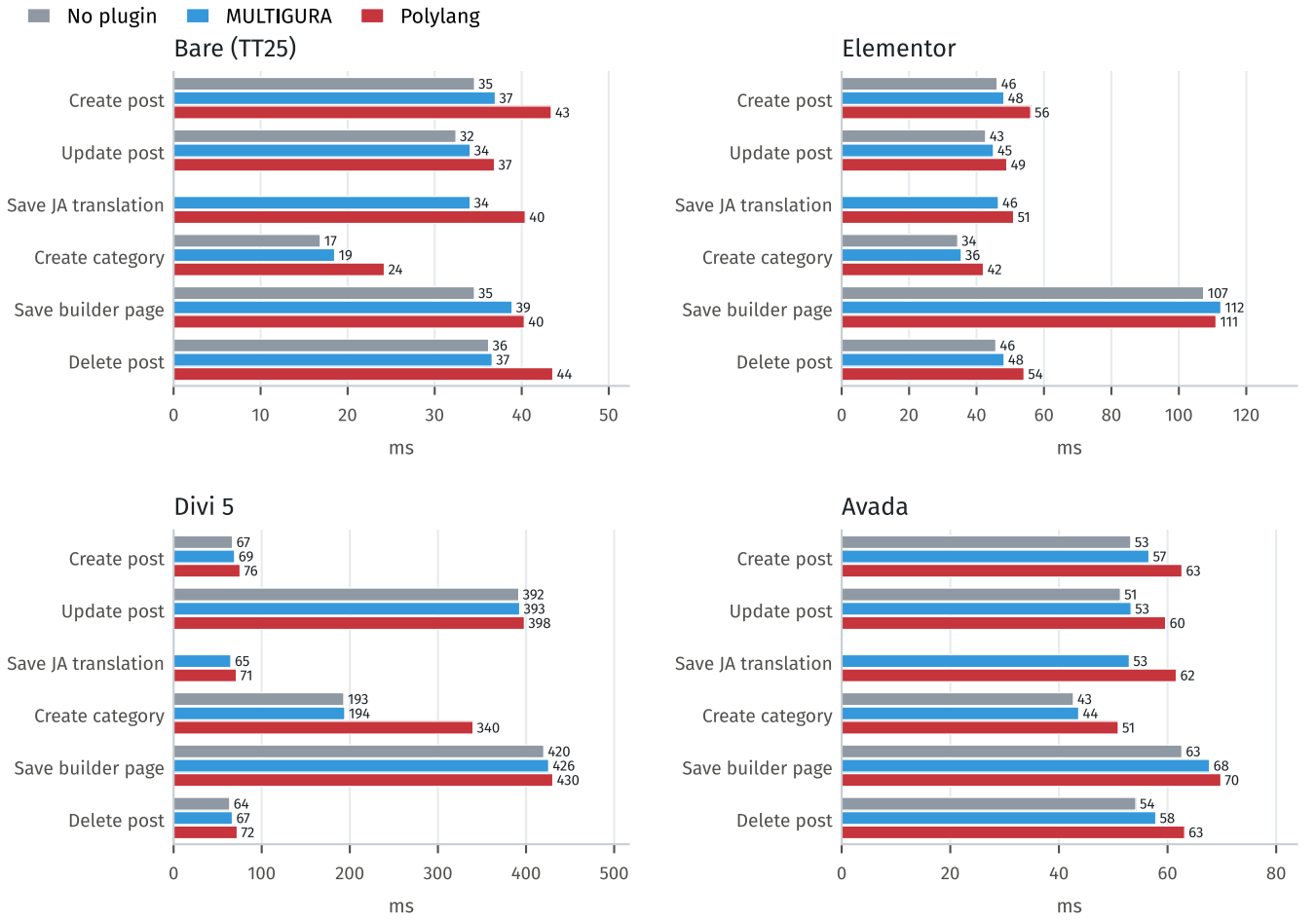

Figure 5. Write requests, server (PHP) time in ms, median (lower is better).

Scenario	Operation	No plugin	MULTIGURA	Polylang
Bare (TT25)	Create post	**35** 103 q	**37** 119 q	**43** 139 q
Bare (TT25)	Update post	**32** 51 q	**34** 65 q	**37** 72 q
Bare (TT25)	Save JA translation	–	**34** 69 q	**40** 86 q
Bare (TT25)	Create category	**17** 26 q	**19** 29 q	**24** 60 q
Bare (TT25)	Save builder page	**35** 43 q	**39** 62 q	**40** 55 q
Bare (TT25)	Delete post	**36** 82 q	**37** 97 q	**44** 111 q
Elementor	Create post	**46** 159 q	**48** 175 q	**56** 195 q
Elementor	Update post	**43** 107 q	**45** 121 q	**49** 128 q
Elementor	Save JA translation	–	**46** 125 q	**51** 142 q
Elementor	Create category	**34** 84 q	**36** 87 q	**42** 117 q
Elementor	Save builder page	**107** 269 q	**112** 310 q	**111** 297 q
Elementor	Delete post	**46** 143 q	**48** 161 q	**54** 172 q

Scenario	Operation	No plugin	MULTIGURA	Polylang
Divi 5	Create post	**67** 149 q	**69** 165 q	**76** 185 q
Divi 5	Update post	**392** 88 q	**393** 102 q	**398** 109 q
Divi 5	Save JA translation	–	**65** 115 q	**71** 132 q
Divi 5	Create category	**193** 50 q	**194** 53 q	**340** 97 q
Divi 5	Save builder page	**420** 97 q	**426** 117 q	**430** 103 q
Divi 5	Delete post	**64** 113 q	**67** 128 q	**72** 142 q
Avada	Create post	**53** 125 q	**57** 141 q	**63** 162 q
Avada	Update post	**51** 76 q	**53** 90 q	**60** 98 q
Avada	Save JA translation	–	**53** 91 q	**62** 109 q
Avada	Create category	**43** 44 q	**44** 47 q	**51** 79 q
Avada	Save builder page	**63** 79 q	**68** 95 q	**70** 89 q
Avada	Delete post	**54** 100 q	**58** 115 q	**63** 130 q

Bulk writes (one WP-CLI process, per item)

100 posts inserted, updated, translated (MULTIGURA, Polylang: a JA post through the plugin's API) and deleted in one PHP process; time and queries per item.

Plugin	Insert	Update	Add JA translation	Delete
No plugin	3.2 ms · 37 q	2.4 ms · 23 q	–	3.0 ms · 37 q
MULTIGURA	4.5 ms · 46 q	4.1 ms · 30 q	6.3 ms · 49 q	3.6 ms · 38 q
Polylang	6.6 ms · 57 q	3.1 ms · 27 q	12.9 ms · 108 q	6.7 ms · 62 q

7. Throughput

Requests per second with 4 concurrent clients (higher is better). The multilingual plugins are shown on the Japanese page, the baseline on its English page.

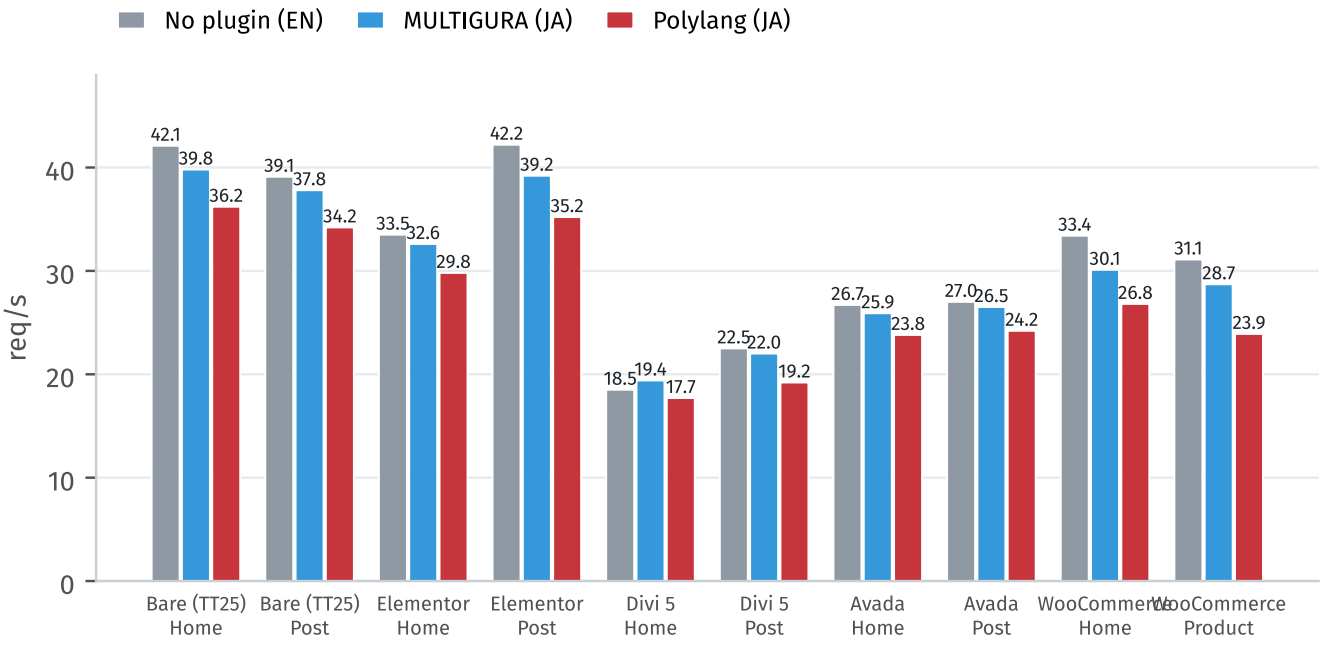

Figure 6. Requests per second with 4 concurrent clients (higher is better).

Scenario	Page	Lang	No plugin	MULTIGURA	Polylang
Bare (TT25)	Home	EN	**42.1** med 74 · p95 84 ms	**40.0** med 78 · p95 89 ms	**37.9** med 83 · p95 94 ms
Bare (TT25)	Home	JA	–	**39.8** med 78 · p95 90 ms	**36.2** med 87 · p95 97 ms
Bare (TT25)	Post	EN	**39.1** med 80 · p95 92 ms	**37.6** med 83 · p95 97 ms	**34.7** med 90 · p95 103 ms
Bare (TT25)	Post	JA	–	**37.8** med 83 · p95 95 ms	**34.2** med 92 · p95 105 ms
Elementor	Home	EN	**33.5** med 104 · p95 135 ms	**32.3** med 110 · p95 126 ms	**29.7** med 119 · p95 135 ms
Elementor	Home	JA	–	**32.6** med 111 · p95 123 ms	**29.8** med 120 · p95 132 ms
Elementor	Post	EN	**42.2** med 75 · p95 85 ms	**40.2** med 79 · p95 89 ms	**35.5** med 90 · p95 100 ms
Elementor	Post	JA	–	**39.2** med 82 · p95 92 ms	**35.2** med 91 · p95 104 ms
Divi 5	Home	EN	**18.5** med 154 · p95 170 ms	**18.2** med 161 · p95 177 ms	**17.2** med 167 · p95 182 ms
Divi 5	Home	JA	–	**19.4** med 160 · p95 180 ms	**17.7** med 180 · p95 194 ms
Divi 5	Post	EN	**22.5** med 137 · p95 148 ms	**21.8** med 142 · p95 158 ms	**20.6** med 150 · p95 168 ms
Divi 5	Post	JA	–	**22.0** med 140 · p95 156 ms	**19.2** med 162 · p95 180 ms
Avada	Home	EN	**26.7** med 116 · p95 144 ms	**25.3** med 122 · p95 153 ms	**23.8** med 130 · p95 155 ms
Avada	Home	JA	–	**25.9** med 121 · p95 143 ms	**23.8** med 132 · p95 165 ms
Avada	Post	EN	**27.0** med 113 · p95 157 ms	**26.8** med 119 · p95 141 ms	**24.2** med 130 · p95 156 ms
Avada	Post	JA	–	**26.5** med 118 · p95 144 ms	**24.2** med 130 · p95 154 ms
WooCommerce	Home	EN	**33.4** med 90 · p95 109 ms	**30.6** med 99 · p95 108 ms	**26.9** med 114 · p95 124 ms
WooCommerce	Home	JA	–	**30.1** med 101 · p95 111 ms	**26.8** med 114 · p95 123 ms
WooCommerce	Product	EN	**31.1** med 98 · p95 111 ms	**28.4** med 108 · p95 116 ms	**24.9** med 122 · p95 131 ms

Scenario	Page	Lang	No plugin	MULTIGURA	Polylang
WooCommerce	Product	JA	–	**28.7** med 106 · p95 116 ms	**23.9** med 128 · p95 138 ms

8. Code and queries per request

What a request loads, mean of the medians over all front-end pages of all scenarios (admin: over all admin screens). PHP files counts every included file (WordPress core, theme, builder and plugins together), so the difference to “No plugin” is the plugin’s own share.

Plugin	PHP files, EN	PHP files, JA	PHP files, admin	Queries, EN	Queries, JA	Memory EN, MB	Memory JA, MB
No plugin	1 124	–	1 105	72	–	10.2	–
MULTIGURA	1 166	1 166	1 149	82	85	10.7	10.7
Polylang	1 372	1 374	1 394	103	104	11.3	12.3

9. Findings

- **Overall.** MULTIGURA was faster than Polylang in 52 of 52 combinations.
- **Japanese pages:** on average MULTIGURA adds +5 ms and Polylang +13 ms over the site without a plugin; Polylang needs 1.13× the time of MULTIGURA.
- **Search:** MULTIGURA adds +3 ms to an English search page over the baseline (content scenarios); home, post, page and category add +4 ms. The search query reads only the posts of the current language.
- **WooCommerce:** MULTIGURA adds +6 ms (EN) / +7 ms (JA) on shop pages, Polylang with Polylang for WooCommerce +15 / +18 ms. The cart: MULTIGURA 71 / 71 ms, Polylang 81 / 86 ms (EN / JA).
- **Queries:** MULTIGURA runs about 10 more queries per English page than WordPress alone (Japanese 13), Polylang about 31 (Japanese 32).
- **Code footprint:** 42 extra PHP files per front-end request with MULTIGURA, 248 with Polylang.

10. Limits

- One machine, Docker Desktop on macOS: absolute numbers differ from a production Linux server; differences between the plugins are reliable to a few milliseconds (interleaved rounds, medians).
- Page caching hides most of these differences for anonymous visitors; they matter for cache misses, logged-in users, the cart and checkout, and search.
- Two languages were measured.
- The shop has simple products only (no variations) and an empty cart for page loads; checkout and payment were not measured.
- Builder editors are JavaScript apps; only their server response is measured.
- Each plugin ran with its defaults plus the settings listed in section 2; Polylang Pro's machine translation (DeepL) was not connected; MULTIGURA's Memcached cache was off.
- The measurement was done by the MULTIGURA team. The scripts and raw results are available on request.