KOSTENLOSE CHECKLISTE · SHOPWARE 6

Die Shopware Performance-Checkliste

15 Quick Wins für einen schnelleren Shop.

Geprüft anhand von Shopware 6.7.14 und der Entwicklerdokumentation, September 2026 · huzaifamustafa.com

Die Standardeinstellungen von Shopware sind so gebaut, dass sie auf jedem Server laufen. In einem Live-Shop mit echtem Traffic verschenken sie damit Geschwindigkeit.

Jeder Punkt unten stammt aus der offiziellen Hosting-Dokumentation von Shopware und ist am Quellcode von 6.7.14 geprüft. Zu jedem finden Sie die Änderung und den Haken. Testen Sie die Punkte zuerst auf Staging, einzeln, und messen Sie vorher und nachher.

Caching

01 HTTP-Cache eingeschaltet lassen

Seiten kommen aus dem Cache, statt bei jeder Anfrage von PHP gerendert zu werden. Für die meisten Shops ist das der größte einzelne Gewinn. Shopware schaltet ihn standardmäßig ein, prüfen Sie also, dass ihn niemand deaktiviert hat.

```
# .env.local
SHOPWARE_HTTP_CACHE_ENABLED=1
```

HAKEN Der eingebaute Cache speichert Seiten im App-Cache. Sobald Sie mehrere App-Server betreiben, gehört ein Reverse Proxy wie Varnish davor.

02 App- und System-Cache nach Redis verschieben

Der Dateisystem-Cache wird unter Last langsam und lässt sich nicht zwischen Servern teilen. Redis kann beides.

```
# config/packages/cache.yaml
framework:
  cache:
    app: cache.adapter.redis_tag_aware
    system: cache.adapter.redis_tag_aware
    default_redis_provider: redis://localhost
```

HAKEN `redis_tag_aware` braucht Shopware 6.5.8.3 oder neuer und die PHP-Redis-Erweiterung. Räumen Sie verwaiste Tags regelmäßig auf (zum Beispiel mit `frosh:redis-tag:cleanup` aus FroshTools), sonst wächst der Speicherverbrauch in Redis weiter.

03 Verzögerte Cache-Invalidierung nach Redis verschieben

Standardmäßig liegen ausstehende Invalidierungen in MySQL. In gut besuchten Shops belastet das die Datenbank zusätzlich.

```
shopware:
  redis:
    connections:
      ephemeral:
        dsn: 'redis://localhost:6379/1'
  cache:
    invalidation:
      delay_options:
        storage: redis
        connection: 'ephemeral'
```

HAKEN `ephemeral` ist ein Verbindungsname und muss unter `shopware.redis.connections` existieren. Lohnt sich bei Shops mit viel Traffic. In einem kleinen Shop merken Sie keinen Unterschied.

04 Warenkorb komprimieren und auf zstd umstellen

Shopware komprimiert Cache-Einträge standardmäßig mit gzip, den Warenkorb aber nicht. Wenn Sie den Warenkorb komprimieren und beides auf zstd umstellen, brauchen Sie weniger Speicher und bewegen weniger Daten zwischen PHP und Storage. zstd ist ab 6.6.4.0 verfügbar.

```
shopware:
  cart:
    compress: true
    compression_method: zstd
  cache:
    compression_method: zstd
```

HAKEN zstd braucht die PHP-Erweiterung zstd, die PHP standardmäßig nicht mitbringt. Installieren Sie sie zuerst (zum Beispiel mit `pecl install zstd` und `extension=zstd` in der php.ini). Leeren Sie den Cache, nachdem Sie die Komprimierungsmethode des Caches geändert haben.

Hintergrundprozesse

05 Admin-Worker abschalten und echte Worker betreiben

Der Admin-Worker verarbeitet die Queue nur, solange jemand die Administration geöffnet hat. Hintergrundjobs bleiben liegen, und der offene Browser-Tab schickt laufend Anfragen an Ihren Server.

```
# config/packages/shopware.yaml
shopware:
  admin_worker:
    enable_admin_worker: false

# mit systemd oder supervisor betreiben
bin/console messenger:consume async low_priority --time-limit=60 --memory-limit=512M
```

HAKEN Betreiben Sie auch einen Worker für den Transport `failed`. Sonst werden fehlgeschlagene Nachrichten nie verarbeitet.

06 Scheduled Tasks per Cron ausführen

Aufräumjobs, Indexierung und andere wiederkehrende Aufgaben laufen pünktlich, statt irgendwann, wenn die Queue dazu kommt.

```
*/5 * * * * /usr/bin/php /var/www/html/bin/console scheduled-task:run --no-wait
```

HAKEN Richten Sie das Cron-Intervall nach Ihrem kürzesten Task-Intervall aus. Alle 5 Minuten passt zu den Standardwerten, denn die Cache-Invalidierung läuft alle 5 Minuten.

07 E-Mails über die Queue versenden

Der Checkout wartet nicht auf den Mailserver. Die Bestellbestätigung geht einen Moment später von einem Worker raus.

```
framework:
  mailer:
    message_bus: 'messenger.default_bus'
```

HAKEN Erst umstellen, wenn die Worker aus Punkt 05 laufen. Sonst bleiben E-Mails in der Queue liegen.

08 Mail-Template-Variablen nicht bei jedem Versand aktualisieren

Jede versendete E-Mail schreibt zusätzlich Beispieldaten in die Datenbank. Wenn Sie das abschalten, fällt ein Schreibvorgang pro Bestellung weg.

```
shopware:
  mail:
    update_mail_variables_on_send: false
```

HAKEN Der Mail-Template-Editor in der Administration verliert seine Beispieldaten für die Autovervollständigung.

09 Sitemap per Cron erzeugen

Bei großen Katalogen ist die Sitemap-Generierung aufwendig. Lassen Sie sie zu einer ruhigen Uhrzeit laufen statt in der Scheduled-Task-Queue. Verfügbar ab 6.7.1.0.

```
shopware:
  sitemap:
    scheduled_task:
      enabled: false

# Cron, zum Beispiel nachts
php bin/console sitemap:generate
```

HAKEN Wenn Sie den Task abschalten und den Cronjob vergessen, wird Ihre Sitemap nicht mehr aktualisiert.

Datenbank und Storefront

10 Reduzierte Produktdaten in Listings laden

Listings laden Produkte ohne die vollständige Beschreibung und die Suchbegriffe. Das senkt Datenbanklast und Speicherverbrauch bei Katalogen mit langen Beschreibungen. Verfügbar ab 6.7.12.0 und in neuen Installationen standardmäßig aktiv. Bestehende Shops müssen es selbst einschalten.

```
Administration: Einstellungen > Produkte >
"Reduzierte Produktdaten in Listings laden"
```

HAKEN Nur aktivieren, wenn Theme und Erweiterungen in Listings weder die vollständige Beschreibung noch Suchbegriffe anzeigen. Prüfen Sie nach dem Einschalten jedes Listing-Template.

11 Product-Stream-Indexer abschalten

Spart Indexierungsarbeit bei jeder Produktänderung, und das summiert sich bei großen Importen schnell. Verfügbar ab 6.6.10.0.

```
shopware:
  product_stream:
    indexing: false
```

HAKEN Kategorieseiten auf Basis einer dynamischen Produktgruppe aktualisieren sich erst, wenn der HTTP-Cache abläuft, und die Regel "Position in dynamischer Produktgruppe" ergibt immer false. Lassen Sie den Punkt aus, wenn Sie diese Regel nutzen.

12 Speculation Rules API aktivieren

Der Browser rendert die Seite vor, die ein Kunde gleich öffnet. Der nächste Klick fühlt sich dadurch sofort an. Verfügbar ab 6.6.10.0.

```
Administration: Einstellungen > System > Storefront >
"Speculation rules API (Experimentell)"
```

HAKEN Das Vorrendern erzeugt zusätzliche Last auf Ihrem Server und kann Ihre Analytics-Zahlen verfälschen.

PHP und Deployment

13 OPcache einstellen und Preloading aktivieren

PHP prüft Dateien nicht mehr bei jeder Anfrage auf Änderungen und hält die Klassen von Shopware im Speicher. Preloading bringt noch einmal 2 bis 5 Prozent.

```
opcache.validate_timestamps=0
opcache.max_accelerated_files=20000
opcache.interned_strings_buffer=20
opcache.enable_file_override=1
opcache.preload=/path/to/var/cache/opcache-preload.php
opcache.preload_user=nginx
zend.assertions=-1
realpath_cache_ttl=3600
```

HAKEN Ohne Timestamp-Prüfung sieht PHP neuen Code nicht. Laden Sie PHP-FPM bei jedem Deployment neu (oder nutzen Sie cachetool). Mit Preloading müssen Sie PHP-FPM außerdem nach jedem Leeren des Caches neu starten, und der Extension Manager funktioniert nicht.

14 Umgebung nach .env.local.php dumpen

Shopware muss die `.env`-Dateien nicht mehr bei jeder Anfrage parsen. Wenn Ihre `APP_URL` stimmt, verhindern Sie außerdem, dass die Administration bei jedem Laden Ihren Shop aufruft, um sie zu prüfen.

```
bin/console system:setup --dump-env
# oder
bin/console dotenv:dump prod

# .env.local, nur wenn APP_URL korrekt ist
APP_URL_CHECK_DISABLED=1
```

HAKEN Führen Sie den Dump erneut aus, sobald sich eine Umgebungsvariable ändert. Die gedumpte Datei hat Vorrang vor Ihren `.env`-Dateien.

15 Autoritative Composer-Classmap nutzen

Der Autoloader schlägt Klassen in einer einzigen Map nach, statt das Dateisystem zu durchsuchen.

```
// composer.json
"config": {
  "optimize-autoloader": true,
  "classmap-authoritative": true
}
```

HAKEN Das lohnt sich nur, wenn alle Plugins über Composer installiert sind. Ein einziges Plugin in `custom/plugins` außerhalb von Composer lässt Shopware den autoritativen Modus wieder abschalten, und der Gewinn ist weg. Führen Sie `composer dump-autoload` bei jedem Deployment aus, damit neue Klassen in der Map stehen.

Hängen Sie bei einem Punkt fest?

Manche Punkte dauern 5 Minuten. Andere, wie Worker und Redis, hängen von Ihrem Hosting ab und können bei falscher Einrichtung etwas kaputt machen. Schreiben Sie mir über das Kontaktformular kurz, worum es in Ihrem Shop geht, und ich antworte per E-Mail.

huzaifamustafa.com/de/kontakt/

Quellen: Shopware 6.7.14.0 Quellcode und Entwicklerdokumentation, developer.shopware.com/docs/guides/hosting/performance/