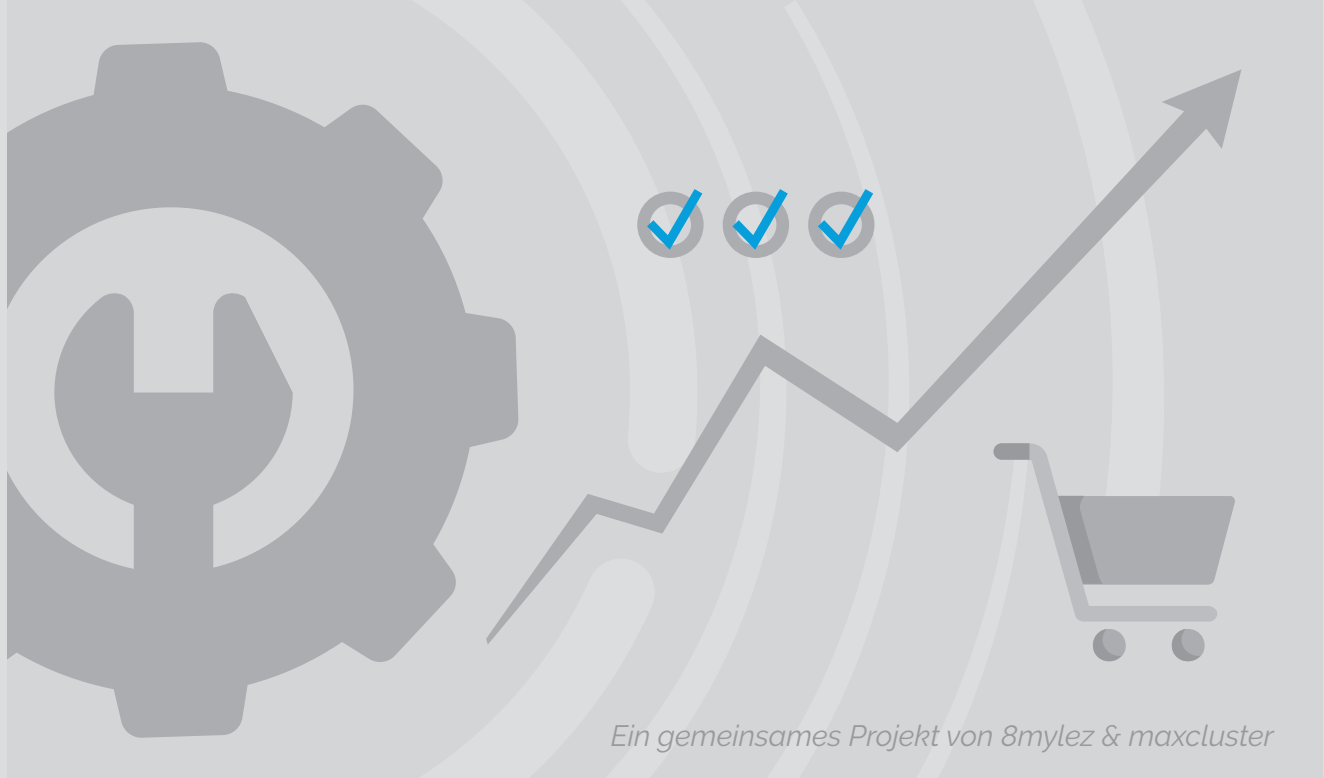




— CASE STUDY

Performance-Grenzen ausreizen

– Shopware-Stresstest für
Bestellungen in Millionenhöhe



CASE STUDY

Inhalt

0 Overview

1. Ausgangssituation: Ein Shop muss Millionen von Bestellungen bewältigen
2. Vorbereitungen: Aufsetzen des Test-Clusters & Performance-Optimierungen
 - Cluster-Setup unter Shopware
 - Performance-Optimierungen
3. Lasttest-Tool & Testablauf
 - Unser individuelles Lasttest-Tool auf Locust-Basis
 - Der Testablauf
4. Ergebnisse: Stabile Performance bei hohem Traffic und großen Datenmengen
5. Herausforderungen & Lösungsansätze: Wo Shopware an Grenzen stößt & was noch verbessert werden muss

6. Fazit



CASE STUDY

Overview

- **11 Mio. Produkte, 230 Mio. Kunden und 160 Mio. Bestellungen**

Stabile Performance auch bei großen Datenmengen:
Der Shopware-Shop bleibt performant, selbst bei Millionenbestellungen und einer enormen Datenbasis.

- **600.000 Requests und 5.000 Bestellungen pro Stunde**

Die Locust-basierte Lasttests von 8mylez simulieren erfolgreich realistische Lastszenarien und hohe Request-Zahlen. Ajax-Nachladungen und kundenspezifisches Caching sorgen für schnelle Ladezeiten, auch bei hohen Nutzerzahlen.

- **Skalierbares Cluster-Setup mit Best-Practices**

Modulare Infrastruktur und Shopware-Best-Practices für maximale Skalierbarkeit.

- **Intelligentes Caching trotz Login & Warenkorb**

Ajax-Nachladungen und kundenspezifisches Caching sorgen für schnelle Ladezeiten, auch bei hohen Nutzerzahlen.

The background of the entire page is a repeating pattern of shopping carts. Some carts are blue with a large white checkmark, while others are grey with a smaller white checkmark. The pattern is scattered across the page, with some carts appearing larger than others.

— CASE STUDY

1. Ausgangssituation: Ein Shop muss Millionen von Bestellungen bewältigen

Ein potenzieller Kunde aus dem Gesundheitsbereich stand vor der Herausforderung, einen Shop mit Millionen von Produkten, Kundendaten und Bestellungen zu migrieren. Besonders wichtig war, dass das System auch bei dieser enormen Datenlast leistungsfähig und zuverlässig bleibt – selbst unter Hochlastbedingungen.

Kann Shopware 6 diese Anforderungen erfüllen? Und wo liegen die technischen Grenzen des Systems, wenn es darum geht, mit so großen Datenmengen und hohem Traffic umzugehen?

Um diese Frage zu beantworten, entschieden wir uns für einen praxisorientierten Testlauf. In Zusammenarbeit mit unserem Hosting-Partner **maxcluster** führten wir umfangreiche Lasttests durch, um die Leistungsfähigkeit von Shopware 6 unter möglichst realen Bedingungen zu überprüfen und herauszufinden, wie gut das System unter extremem Daten- und Traffic-Volumen arbeitet.

2. Vorbereitungen: Aufsetzen des Test- Clusters & Performance- Optimierungen

Cluster-Setup unter Shopware

Für unseren Test setzten wir auf ein modulares Cluster-Setup, bei dem spezialisierte Server jeweils dedizierte Aufgaben übernehmen – etwa Loadbalancing, Caching, Datenbankzugriffe oder Queue-Handling.

Ein zentraler Loadbalancer verteilt die Anfragen über Varnish an mehrere Anwendungsserver, die jeweils den vollständigen Shopware-Code samt aller relevanten Konfigurationen enthalten. Die Infrastruktur wird durch leistungsfähige Dienste ergänzt:

- *Redis für Session-Handling, Caching, Warenkörbe und Locking*
- *RabbitMQ für Message-Queues*
- *Opensearch für eine performante Suche*
- *ein Master/Slave-Datenbank-Setup zur Optimierung von Lese- und Schreibzugriffen*

Gemeinsam genutzte Daten wie Medien oder generierte Assets werden über ein Shared Filesystem bereitgestellt.

Die Konfiguration folgt den Shopware Best Practices, z. B. dem Einsatz von Redis in verschiedenen Bereichen sowie Varnish als Full Page Cache – inklusive Workarounds für aktuelle Einschränkungen bei xkey.

Die Einrichtung erfolgte zunächst auf einem einzelnen Anwendungsserver. Danach wurde der Shop per Deployment und Dateisynchronisierung auf weitere Server verteilt. Zusätzlich kamen interne Performance-Plugins wie EmzPerformance zum Einsatz, um die Anwendungsebene gezielt zu optimieren.

Wir haben die wichtigsten Cluster-Settings abgestimmt – etwa Redis für Sessions – und den kompletten Cluster-Workflow so konfiguriert, dass die Performance auch bei starker Auslastung stabil bleibt. Dadurch konnten wir die Cache-Hit-Raten spürbar erhöhen und so eine hohe Geschwindigkeit und Zuverlässigkeit auch bei hohem Traffic sicherstellen.

Performance-Optimierungen

Für den Einstieg in die Optimierung diente uns die offizielle Performance-Checkliste von Shopware als hilfreiche Grundlage. Darauf aufbauend haben wir gezielt weitere Maßnahmen aus unserem eigenen Erfahrungsschatz ergänzt.

Im Rahmen des Tests setzten wir unser mittleres Performance-Paket ein – eines von drei Optimierungspaketen, die wir unseren Kunden anbieten. Der Fokus dieses Pakets

liegt insbesondere auf der Verbesserung des Standard-Cachings von Shopware.

Da sich das Standardverhalten des Caches in einigen Bereichen aus unserer Sicht nicht ideal verhält, haben wir gezielte Anpassungen vorgenommen, um die Caching-Effizienz zu steigern und das System noch robuster für hohe Lasten zu machen.



Cache-Optimierungen:

Unser Ziel: So viele Seiten wie möglich aus dem Cache ausliefern, um eine hohe Performance und Stabilität auch bei starkem Traffic sicherzustellen.

Im Standardverhalten von Shopware wird der Cache oft umgangen – etwa sobald sich ein Kunde einloggt oder Produkte in den Warenkorb legt. Das führt dazu, dass Seiten unnötig dynamisch generiert werden und die Serverlast steigt.

Unsere Optimierungen setzen genau hier an:

- Auch eingeloggte Nutzer profitieren weiterhin vom Cache, da wir gezielt zwischen statischen und individuellen Inhalten unterscheiden.
- Kundenspezifische Inhalte wie personalisierte Preise oder Warenkörbe werden über Ajax nachgeladen oder mithilfe cachegetrennter Kundengruppen ausgeliefert.
- Zusätzlich haben wir die Cache-Invalidierungslogik überarbeitet, da diese im Standard zu aggressiv agiert und Seiten zu häufig aus dem Cache entfernt werden.

Mit diesen Maßnahmen bleibt der Shop auch bei hohen Nutzerzahlen performant – bei gleichzeitig reduzierter Serverlast.

PHP- und Datenbank-Optimierungen:

Ergänzend zu den Cache-Anpassungen haben wir auch die PHP- und Datenbankkonfiguration gezielt auf Performance ausgerichtet:

- Die PHP-Version wurde überprüft und die PHP-Settings entsprechend abgestimmt.
- Der PHP-OPCache wurde feinjustiert, um eine möglichst effiziente Ausführung des Codes zu gewährleisten.
- Auf Datenbankebene haben wir den InnoDB Buffer Pool optimiert und die MySQL-Version geprüft, um auch bei großen Datenmengen eine hohe Verarbeitungsgeschwindigkeit sicherzustellen.

Diese Maßnahmen sorgen für ein performantes Zusammenspiel von Anwendung und Datenbank – besonders wichtig bei wachsendem Traffic und komplexen Shopstrukturen.

3. Lasttest-Tool & Testablauf

Unser individuelles Lasttest-Tool auf Locust-Basis

Die Lasttests wurden in mehreren Stufen mit konstantem Traffic durchgeführt – dabei simulierten wir bis zu 600.000 Requests und 5.000 Bestellungen pro Stunde. Unser Fokus lag dabei ausschließlich auf dem Backend und den Serverprozessen. Elemente wie Bilder, JavaScript oder andere Frontend-Komponenten haben wir bewusst ausgeklammert, da diese erfahrungsgemäß kein typisches Bottleneck bei hoher Last darstellen.

Als technische Basis diente Locust, ein bekanntes, Python-basiertes Framework für Lasttests. Locust bildet allerdings nur das Grundgerüst – die konkrete Simulation von Nutzerverhalten (z. B. Produkte in den Warenkorb legen) muss individuell definiert werden.

Daher entwickelten wir ein eigenes Lasttest-Tool, das auf einem Open-Source-Tool von Tideways basiert, welches

wiederum auf Locust aufbaut. Das ursprüngliche Tideways-Tool überzeugte uns jedoch nicht vollständig in der Bedienung. Deshalb haben wir es als Grundlage genutzt und mit Shopware-spezifischen Locust-Logiken erweitert. So entstand ein flexibles Tool, das genau auf unsere Anforderungen zugeschnitten ist und inzwischen auch intern für andere Kundenprojekte zum Einsatz kommt.

Mit unserem Tool lassen sich Kundentypen, Nutzeranzahl, Testdauer und Szenarien frei konfigurieren – inklusive der Gewichtung einzelner Aktionen. In unseren Tests simulierten wir 125 gleichzeitige Besucher über eine Laufzeit von einer Stunde. Getestet wurde ausschließlich Kundentraffic, API-Importe waren nicht Bestandteil – könnten aber bei Bedarf problemlos ergänzt werden.



Der Testablauf

So sind wir bei den Lasttests konkret vorgegangen:

Zunächst haben wir auf dem Testcluster von maxcluster eine Shopware-Instanz ohne Daten aufgesetzt.

Phase 1

Anschließend wurden 100.000 Artikel sowie erste Kunden- und Bestelldaten importiert.

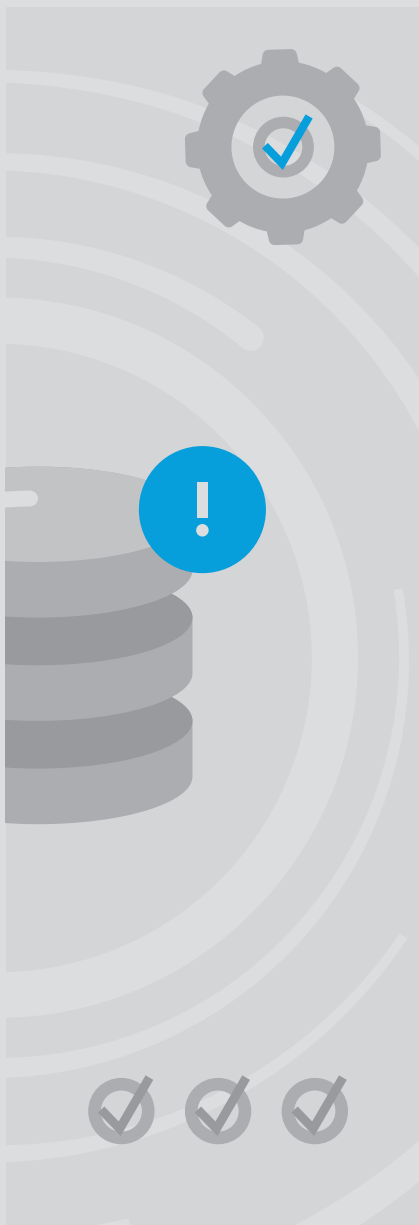
Phase 2

Die Artikelanzahl wurde verdoppelt, und wir haben jeweils 4.000.000 Kunden- und Bestelldatensätze eingespielt.

Phase 3

Die Datenmenge stieg weiter auf 600.000 Artikel sowie rund 15.000.000 Kunden und 14.000.000 Bestellungen.

Fazit In allen drei Testphasen blieb der Shopware-Shop stabil und performant.



Darüber hinaus haben wir den Shop manuell weiter getestet und die Datenbasis deutlich erhöht: auf 11.000.000 Produkte, 230.000.000 Kunden und 160.000.000 Bestellungen. Auch unter diesen Bedingungen blieb das System stabil.

Mit unserem Tool lassen sich Kundentypen, Nutzeranzahl, Testdauer und Szenarien frei konfigurieren – inklusive der Gewichtung einzelner Aktionen. In unseren Tests simulierten wir 125 gleichzeitige Besucher über eine Laufzeit von einer Stunde. Getestet wurde ausschließlich Kundentraffic, API-Importe waren nicht Bestandteil – könnten aber bei Bedarf problemlos ergänzt werden.

Unser Hauptziel war dabei stets, sicherzustellen, dass der Shop trotz hoher Datenlast zuverlässig erreichbar und performant bleibt – insbesondere im Hinblick auf die Serverantwortzeiten wie TTFB (Average und 95th Percentile). Zwar haben wir keine exakten Schwellenwerte definiert, aber wir haben genau beobachtet, wie viele Requests und Bestellungen pro Sekunde über eine Stunde hinweg verarbeitet werden konnten, ohne dass die Performance spürbar abnimmt.

— CASE STUDY

4. Ergebnisse: Stabile Performance bei hohem Traffic und großen Datenmengen

Unsere Lasttests haben gezeigt: Die Anzahl der Requests und Bestellungen pro Sekunde blieb über eine Stunde hinweg konstant. Auch die Serverantwortzeiten veränderten sich kaum – im Durchschnitt lagen sie bei etwa 0,8 Sekunden, das 95th Percentile bei rund 3,6 Sekunden. Unsere wichtigsten Learnings:

- *Die maximale Anzahl an gleichzeitigen Requests und Bestellungen pro Sekunde hängt maßgeblich von den verfügbaren Ressourcen im Cluster ab.*
- *Die Infrastruktur von maxcluster hat hier überzeugt: performant, zuverlässig und flexibel erweiterbar. Dank enger Zusammenarbeit mit unseren Ansprechpartnern konnten wir das Set-up effizient umsetzen – inklusive technischer Beratung bei Konfiguration und Optimierung.*
- *Die Architektur von Shopware lässt eine unkomplizierte horizontale Skalierung zu: Bei wachsendem Traffic können einfach zusätzliche App-Server angebunden werden.*



Zwischenfazit

Große Datenmengen bringen Shopware 6 im Standard zunächst nicht aus dem Gleichgewicht. Weder die Registrierung neuer Kunden noch das Platzieren von Bestellungen wird bei hohen Datenvolumina – selbst bei über 200 Millionen Datensätzen – zum Flaschenhals.

5. Herausforderungen & Lösungsansätze: Wo Shopware an Grenzen stößt & was noch verbessert werden muss

Die eigentlichen Herausforderungen entstehen weniger durch Shopware selbst, sondern durch technologische Limitierungen – etwa bei MySQL oder in angrenzenden Systemen – sobald sehr große Datenmengen verarbeitet werden müssen. Für die folgenden Bereiche müssen wir noch gezielte, maßgeschneiderte Ansätze entwickeln:

Produktindexierung in Elasticsearch:

Der Indexierungsprozess dauert derzeit zu lange und wird mit wachsendem Datenvolumen zunehmend zum Flaschenhals.

Performance im Adminbereich:

Der Shopware-Standard reicht für eine reibungslose Arbeit im Admin bei großen Datenmengen nicht aus. Es existieren zwar Plugins, diese sind aber meist nicht für größere Kundenprojekte geeignet.

Backup & Restore-Prozesse:

Das Sichern und Wiederherstellen der Datenbanken nehmen viel Zeit in Anspruch. Eine mögliche Lösung könnte ein Archivierungskonzept sein, bei dem ältere, selten genutzte Bestellungen oder Kundendaten extern gespeichert und nur bei Bedarf wieder geladen werden.

Migrationen & Updates:

Datenbankmigrationen und Systemupdates werden bei großen Datenmengen komplexer – hier sind gezielte Strategien notwendig.

Statistiken & Auswertungen:

Aktuell durchsucht Shopware für Statistiken alle Datensätze, was bei Millionen von Einträgen zu spürbaren Verzögerungen führt. Denkbar wäre hier eine Integration mit Elasticsearch oder der Einsatz externer Auswertungstools.

Caching & Varnish:

Es kann zu Problemen kommen, wenn gecachte Requests nicht korrekt an die App-Server weitergeleitet werden. Hier ist besondere Vorsicht bei Varnish-Konfigurationen geboten.

Automatisierte Lasttests während der Entwicklung:

Wir planen, regelmäßige Lasttests schon während der Entwicklungsphase automatisiert durchzuführen. So lassen sich Engpässe, die durch individuelle Anpassungen entstehen (und nicht im Shopware-Standard enthalten sind), frühzeitig erkennen und beheben.



6. Fazit

Die durchgeführten Tests mit Shopware 6, in Zusammenarbeit mit unserem Hosting-Partner maxcluster, bestätigen die herausragende Skalierbarkeit und Performance des Systems. Sowohl bei extrem hohem Traffic als auch bei gigantischen Datenmengen bleibt die Plattform stabil und leistungsfähig.

- **Skalierbarkeit:** Auch bei 5.000 Bestellungen pro Stunde und konstantem Traffic von bis zu 125 gleichzeitigen Besuchern zeigt Shopware 6 keinerlei Performanceeinbrüche. Das flexible Cluster-Setup ermöglicht es, je nach Bedarf Server hinzuzufügen oder zu entfernen – ein großer Vorteil, insbesondere während Spitzenzeiten wie dem Black Friday.
- **Datenmengen:** Selbst bei Millionen von Bestellungen und Produkten bleibt Shopware 6 leistungsstark. In unseren Tests haben wir problemlos bis zu 600.000 Produkte und 15 Millionen Bestellungen verarbeitet. Der Einsatz von Elasticsearch und optimierten Caching-Mechanismen gewährleistet, dass die Performance bei großen Katalogen und komplexen Suchanfragen nicht beeinträchtigt wird.
- **Cluster-Infrastruktur:** Das durchdachte Cluster-Setup, das verschiedene Server für Loadbalancing, Caching (Redis), Queue-Handling (RabbitMQ) und schnellen Datenzugriff (OpenSearch, Master/Slave-Datenbanken) umfasst, bildet die Grundlage für die hohe Performance von Shopware 6. Die Tests zeigen, dass mit der richtigen Infrastruktur nahezu jede Herausforderung gemeistert werden kann.

Insgesamt zeigt sich: Shopware 6 ist nicht nur für kleinere Shops geeignet, sondern kann auch bei großen, skalierenden E-Commerce-Projekten mit hohem Traffic und großen Datenmengen überzeugen. Die Möglichkeit zur flexiblen Skalierung, gepaart mit leistungsstarker Infrastruktur, bietet eine ausgezeichnete Grundlage für zukunftssichere Online-Shops.

Ein besonderer Dank geht an unseren Partner maxcluster, der uns mit der Bereitstellung des Testclusters und seiner Expertise maßgeblich bei der Durchführung dieser Tests unterstützt hat.