

مقارنة) M Users إلى 5 K من 1 Scaling Laravel :محاضرة عملية

فكرة مهمة قبل أي حاجة 0)

نفس الكود ممكن يشتغل في الحالتين... بس اللي بيختلف جامد هو

- المعمارية (Architecture)
- الداتا بيز (DB Design + Indexing + Sharding/Partition)
- الكاش (Caching Strategy)
- المعالجة الخلفية (Queues + Jobs)
- الديبلويمنت (Deployment + Infra)
- المراقبة (Observability)
- التحكم في التكلفة (Cost Control)

المتطلبات والأهداف 1)

عند 1000 مستخدم

- "هدفك: "يشتغل كويس" + "تطوير سريع"
- تقدر تعتمد على:
 - سيرفر واحد أو اثنين
 - واحدة DB
 - كاش بسيط أو حتى من غيره
 - Cron بسيط أو Queue

عند 5,000,000 مستخدم

- "هدفك: "يشتغل دايماً" + "يتحمل الضغط" + "مفيش توقف" + "زمن استجابة ثابت"
- لازم تفكر في:
 - High Availability
 - Horizontal scaling
 - Read/Write separation
 - Caching layers
 - Queue at scale
 - Zero-downtime deploy
 - Monitoring + Alerting
 - Rate limiting + Abuse protection

2) السعة (Capacity) vs الأداء (Performance)

1K

- قليلة نسبيًا Requests
- زيادة مش هتبان قوي Query أي
- وممكن تعدي Query أي صفحة ممكن تعمل 5-20

5M

- زيادة = كارثة Query أي
- بيدمّر الدنيا N+1
- لازم:
 - eager loading صح
 - pagination في كل listing
 - منع تحميل علاقات مالهاش لازمة
 - تجنب الحقول الثقيلة + (API responses) payload ضغط ال

قاعدة البيانات: أكبر فرق (3)

1K Users

- واحدة MySQL
- Index بسيط على (email, user_id, created_at)
- Backups بسيطة
- Locks مفيش ضغط كبير على

“لازم تخطط للداتا كأنها “منتج لوحدها 5M Users

أ) Indexing و Query Discipline

- لازم يبقى production في Query كل
 - محدد (select columns)
 - مناسب index عليه
 - avoid full table scans
 - Composite indexes (مثلاً user_id, status, created_at)

ب) Read Replicas

- فصل قراءة عن كتابة:
 - Writes على primary
 - Reads على replicas
 - بسهولة config يدعمها في Laravel

ج) Partitioning / Archiving

- بتنفجر حجمها logs, events, transactions جداول
- تعمل:
 - partition by date
 - لجدول تاريخي archive أو

locks إدارة ال

- غلط على صفوف كثير يوقف الدنيا M، عند 5
- لازم:
 - معاملات قصيرة
 - في أوقات الذروة “update all” avoid
 - للعمليات الكبيرة queues استخدام

4) "من اختياري" إلى "أساسي" (Caching): الكاش

1K

- config كاش بسيط للـ
- صغير Redis أو file cache ممكن تستخدم

5M

- بوضوح، والكاش يبقى طبقات (Memcached أو Redis) لازم

أنواع كاش مهمة:

- Response/HTTP cache (لو مناسب)
- Query Result Cache (expensive queries نتائج)
- Object Cache (user profile, settings, permissions)
- Rate Limit Cache (حماية)

قواعد ذهبية:

- Cache invalidation strategy واضحة
- TTL مناسب
- Cache keys منظمة

5) Background Jobs & Queues: M قلب النظام في 5

1K

- Cron أو sync ممكن تعمل إرسال الإيميل/الإشعارات

5M

- Queue: كل حاجة "ثقيلة" لازم تروح
 - emails / SMS / push
 - report generation
 - imports/exports
 - image processing
 - webhooks retries
 - analytics events

Laravel tooling مهم:

- Redis Queue
- Laravel Horizon لمراقبة workers
- priority: حسب queues تقسيم ال
 - high, default, low

6) Auth الجلسات والـ

1K

- تمثلي (file/session driver) على السيرفر Sessions

5M

- لو عندك أكثر من سيرفر:
 - sessions لازم تبقى shared:
 - Redis session driver
 - tokens/JWT لبعض الأنظمة
- لازم تفكر في:
 - password reset throttle
 - 2FA (لو منتج حساس)
 - device/session management

7) والبنية التحتية Deployment الـ 7)**1K**

- VPS واحد + Nginx + PHP-FPM
- Deploy يدوي أو Git pull

5M

- Load balancer + عدة app servers
- Auto-scaling (حسب CPU/RPS)
- Zero-downtime deploy (Blue/Green أو Rolling)
- Separate services:
 - App
 - DB
 - Redis
 - Queue workers
 - Scheduler
- CDN للأصول (assets/images)

8) Observability: "من" لو وقع هنعرف" إلى "لازم نعرف قبل ما يقع"**1K**

- Logs بسيطة
- server metrics شوية

5M

- لازم:
 - Centralized logging
 - APM (traces)
 - Metrics + Alerting
 - SLO/SLA targets

أمثلة مؤشرات تراقبها:

- p95 latency
- error rate

- DB slow queries
- queue lag
- cache hit ratio
- CPU/memory per node

9) الأمان والحماية من الإساءة (Abuse)

1K

- validation + auth + basic rate limit

5M

- Rate limiting قوي على:
 - login
 - OTP
 - search endpoints
- WAF / bot protection (حسب نوع المنتج)
- منع scraping
- permissions caching
- audit logging للعمليات الحساسة

10) Laravel-specific: حاجات بتفرق جدًا

- `php artisan config:cache` و `route:cache` في production
- Redis استخدام:
 - cache
 - sessions
 - queues
- Horizon لـ workers
- Octane (throughput لو مناسب) لتحسين
- Database pooling/connection tuning
- debug tools في production منع

(مقارنة مختصرة "على شكل جدول" بالمعنى)

1K Users

- Single server possible
- DB واحدة
- قليل كاش
- قليلة jobs
- بسيط deploy
- بسيط monitoring

5M Users

- multi-server + load balancer
- DB primary + replicas + partitioning
- طبقات caching

- queue heavy + priorities
- CI/CD + rolling deploy
- APM + alerts + SLOs

Roadmap 1 K → 5M عملي: لو مشروعك هيكبر من 1

1. Fix DB queries + indexing + pagination
2. Redis caching
3. Queues + Horizon
4. Read replicas
5. CDN + optimize assets
6. Observability
7. Auto-scaling + zero-downtime deploy
8. Partition/Archive big tables