13

O‘N UCHINCHI AMALIY MASHG‘ULOT

Korxona ilovalarini dasturlash · 2 soat · 13-mavzu

Profillash, indekslar va keshlash

Jamoa loyihasi bosqichi · 5 ta PHP topshiriq · o‘qituvchi bahosi

N+1 | Indekslar | Redis kesh | Loglash | Sentry

MASHG‘ULOT MAQSADI

Amaliy mashg‘ulot oxirida siz quyidagilarni bajara olasiz...

1. **O‘lchash** — Debugbar va so‘rovlar logi yordamida asosiy sahifalarning boshlang‘ich ko‘rsatkichlarini
2. **Bartaraf etish** — buyurtmalar, mahsulotlar va hisob-fakturalar sahifalaridagi N+1 muammolarini
3. **Qo‘shish** — haqiqiy so‘rovlar uchun indekslarni va ularni `EXPLAIN ANALYZE` bilan isbotlashni
4. **Keshlash** — hisobotlarni Redis’da teglar bilan va hodisa bo‘yicha invalidation bilan
5. **Sozlash** — JSON loglar, `request_id`, `/up` va `/health` endpointlarini
6. **Yozish** — “oldin/keyin” raqamlari bilan unumdorlik hisobotini (`docs/performance.md`)

Qanday bajariladi va qanday baholanadi?

Ish tartibi

1. Saytga login va parolingiz bilan kiring: **emro.uz/korxona/kirish.html**
2. **13-amaliy** sahifasini oching (oxirgi slayddagi QR kod).
3. Yo‘riqnoma bo‘yicha jamoa loyihasida bosqichni bajaring, **commit** va **push** qiling.
4. PHP topshiriqlarni yeching: ▶ **Sinab ko‘rish** → ✓ **Topshirish** (yashirin testlar).
5. Loyiha havolasini (GitHub repo, PR yoki tag) topshiring — o‘qituvchi **baho** qo‘yadi.

Baholash mezonlari

- **Natija** — loyiha bosqichi mezonlar ro‘yxati bo‘yicha bajarilgan.
- **Topshiriqlar** — PHP topshiriqlar testlardan o‘tgan.
- **Kod sifati** — tushunarli nomlar, PSR-12, kichik va mazmunli commitlar.
- **Jamoaviy hissa** — har bir a’zo o‘z qismini tushuntira oladi.
- **Muddat** — muddatdan keyin topshirilgan ish “kechikkan” deb belgilanadi.

01

1-bosqich: profillash va N+1 ni bartaraf etish

Katta test ma’lumotlari · Debugbar · eager loading · so‘rovlar soni testi

Debugbar | Seeder | with() | Pest

1-QADAM

Ma’lumot va profiler

```
composer require barryvdh/laravel-debugbar --dev
php artisan make:seeder PerformanceSeeder
php artisan migrate:fresh --seed --seeder=PerformanceSeeder

# so'rovlar sonini tez tekshirish (tinker)
php artisan tinker
>>> DB::enableQueryLog();
>>> Order::latest()->take(50)->get()->each->counterparty;
>>> count(DB::getQueryLog());
```

▶ NATIJA

```
Database\Seeders\PerformanceSeeder ..... 48s DONE
= 51
```

`migrate:fresh`

faqat lokal bazada! barcha jadvallarni o‘chirib qayta yaratadi

`--seeder=`

faqat katta hajmli seeder ishga tushadi

`->each->counterparty`

har bir buyurtma uchun lazy so‘rov — 51 ta

O‘LCHOV JADVALI

Boshlang‘ich nuqta (baseline) — namuna

Sahifa	So‘rovlar (oldin → keyin)	Vaqt (oldin → keyin)	Nima qilindi
`/orders` **(50 ta)**	103 → 6	2,4 s → 180 ms	`with(['counterparty', 'warehouse'])`, `withCount('items')`
`/orders/{id}`	38 → 4	420 ms → 60 ms	`load('items.product', 'invoice.payments')`
`ProductTable` **(Livewire)**	62 → 3	900 ms → 110 ms	`with('category')`, stocks yig‘indisi `withSum()`
`GET /api/v1/products`	41 → 3	350 ms → 70 ms	`with()` + `whenLoaded()` resursda
Dashboard	12 → 12	3,1 s → 3,1 s	N+1 emas — og‘ir agregatsiya: 2-bosqichda kesh

Raqamlar — namuna. O‘z loyihangizda haqiqiy o‘lchovlarni yozing; bir xil ma’lumot va bir xil muhitda o‘lchang.

REGRESSIYA TESTI

So‘rovlar soni buyurtmalar soniga bog‘liq emas

```php
<?php

use App\Models\Order;
use App\Models\User;
use Illuminate\Support\Facades\DB;

it('orders sahifasida N+1 yo‘q', function () {
    $this->actingAs(User::factory()->create());

    $count = function (int $n): int {
        Order::factory()->count($n)->hasItems(3)->create();
        DB::flushQueryLog();
        DB::enableQueryLog();
        $this->get('/orders')->assertOk();
        return count(DB::getQueryLog());
    };
    expect($count(5))->toBe($count(25));
});
```

▶ NATIJA

```
PASS  Tests\Feature\OrdersPerformanceTest
✓ orders sahifasida N+1 yo‘q                0.84s
```

`$count(5) vs $count(25)`

N+1 bo‘lsa, ikkinchi son katta chiqadi va test yiqiladi

`DB::flushQueryLog()`

ma’lumot tayyorlash so‘rovlari hisobga kirmaydi

`hasItems(3)`

factory’ning sehrli `has{Munosabat}` metodi: har buyurtmaga 3 ta pozitsiya

Ikkinchi chaqiruvda bazada 30 ta buyurtma bo‘ladi — bu atayin: hajm oshsa ham so‘rovlar soni o‘zgarmasligi kerak.

02

2-bosqich: indekslar va hisobotlarni keshlash

So‘rovlarni yig‘ish · indeks migratsiyasi · EXPLAIN ANALYZE · Redis kesh va invalidation

Indeks | EXPLAIN ANALYZE | Redis | Cache::tags

INDEKS MIGRATSIYASI

add_performance_indexes

```php
public function up(): void
{
    Schema::table('orders', function (Blueprint $table) {
        $table->index(['status', 'created_at']);
        $table->index('counterparty_id');
    });
    Schema::table('stock_movements', function (Blueprint $table) {
        $table->index(['product_id', 'created_at']);
    });
    Schema::table('invoices', function (Blueprint $table) {
        $table->index(['status', 'due_at']);
    });
}
public function down(): void
{
    Schema::table('orders', fn (Blueprint $t) => $t->dropIndex(
        ['status', 'created_at']));
}
```

▶ NATIJA

```
INFO  Running migrations.
2026_10_27_120000_add_performance_indexes ..... 1.2s DONE
```

`index([...])`

har bir indeks — sekin so‘rovlar ro‘yxatidagi aniq so‘rov uchun

`dropIndex([...])`

ustunlar massivi bo‘yicha Laravel indeks nomini o‘zi tiklaydi

`down()`

namuna qisqartirilgan: har bir `index()` uchun `dropIndex()` yozing

ISBOT

EXPLAIN (ANALYZE, BUFFERS): oldin va keyin

```sql
-- muddati o'tgan to'lanmagan hisob-fakturalar
EXPLAIN (ANALYZE, BUFFERS)
SELECT id, number, total, paid_amount
FROM invoices
WHERE status = 'unpaid' AND due_at < '2026-10-27'
ORDER BY due_at
LIMIT 100;

-- migratsiyadan keyin statistikani yangilash
ANALYZE invoices;
```

▶ NATIJA

```
-- oldin:  Seq Scan on invoices ... Rows Removed by Filter: 118930
--         Buffers: shared hit=2210   Execution Time: 28.7 ms
-- keyin:  Index Scan using invoices_status_due_at_index on
invoices
--         Buffers: shared hit=9      Execution Time: 0.3 ms
```

BUFFERS

nechta sahifa (8 KB) o‘qilganini ko‘rsatadi — vaqtdan barqarorroq ko‘rsatkich

ANALYZE invoices

planner statistikasini yangilaydi

oldin/keyin

ikkala reja ham `docs/performance.md` ga qo‘shiladi

REDIS

Dashboard keshi va invalidation

```php
public function dashboard(int $whId): array
{
    return Cache::tags(['reports', "warehouse:{$whId}"])->remember(
        "report:dashboard:v1:wh={$whId}",
        now()->addMinutes(5),
        fn () => [
            'today_orders' => $this->todayOrders($whId),
            'low_stock' => $this->lowStockCount($whId),
            'unpaid_total' => $this->unpaidTotal(),
        ],
    );
}

// app/Listeners/FlushReportCache.php
public function handle(OrderConfirmed $event): void
{
    Cache::tags(["warehouse:{$event->order->warehouse_id}"])->flush();
}
```

▶ NATIJA

```
GET /dashboard  1-marta: MISS  12 SQL  3,1 s
GET /dashboard  2-marta: HIT    0 SQL  95 ms
OrderConfirmed → flush → 3-marta: MISS
```

v1

hisobot formati o‘zgarsa `v2` — eski kesh avtomatik ishlatilmay qoladi

now()->addMinutes(5)

invalidation unutilsa ham ma’lumot 5 daqiqadan ortiq eskirmaydi

warehouse:{id}

faqat o‘zgargan omborning keshi tozalanadi

03

3-bosqich: loglash, health endpoint va unumdorlik hisoboti

JSON kanal va request_id · /up va /health · Sentry (ixtiyoriy) · docs/performance.md

JSON log | /health | Sentry | Hisobot

SINOV

Health endpointni tekshirish

```
curl -i https://omborpro.test/up
curl -s https://omborpro.test/health | jq .status

docker compose stop db
curl -s -o /dev/null -w "%{http_code}\n" https://omborpro.test/up
curl -s -o /dev/null -w "%{http_code}\n" https://omborpro.test/health
docker compose start db

# loglarda parol yo'qligini tekshirish
grep -ri "password" storage/logs/ || echo 'toza'
```

▶ NATIJA

```
HTTP/1.1 200 OK
"ok"
500
503
toza
```

`/up` → `500`

`DiagnosingHealth` listener'idagi `DB::select` istisno tashladi

`/health` → `503`

muhim tekshiruv (`database`) ishlamadi — `fail`

`grep ... || echo`

hech narsa topilmasa "toza" chiqadi

HISOBOT

docs/performance.md tuzilmasi

1

Muhit

Docker stek, PostgreSQL 16, Redis 7; ma’lumot hajmi (20 000 buyurtma va h.k.)

2

Usul

Debugbar, `DB::getQueryLog()`, `EXPLAIN (ANALYZE, BUFFERS)`, 5 marta o‘lchab mediana

3

N+1

sahifalar jadvali: so‘rovlar va vaqt oldin → keyin, nima o‘zgartirildi

4

Indekslar

har bir indeks: qaysi so‘rov uchun, reja oldin/keyin

5

Kesh

kalitlar, TTL, teglar, invalidation hodisalari; HIT/MISS isboti

6

Kuzatuvchanlik

JSON log namunasi, `request_id`, `/up` va `/health` javoblari, Sentry (bo‘lsa)

Hisobot oxirida: “hali hal qilinmagan” bo‘limi — halol yozilgan cheklovlar ijobiy baholanadi.

LOYIHA BOSQICHI

Natija va topshirish

Natija: asosiy sahifalarda N+1 muammolari bartaraf etilgan, haqiqiy so‘rovlar uchun indekslar qo‘shilgan, hisobotlar Redis’da keshlanadi, JSON loglar va health endpoint ishlaydi; “oldin/keyin” raqamlari bilan unumdorlik hisoboti tayyor.

Baholash mezonlari

- ✓ `PerformanceSeeder` mavjud va kamida 10 000 ta buyurtma yaratadi; Debugbar faqat `require-dev` da.
- ✓ Kamida 3 ta sahifa yoki endpoint uchun N+1 bartaraf etilgan; `Model::preventLazyLoading()` dev muhitda yoqilgan.
- ✓ So‘rovlar soni buyurtmalar soniga bog‘liq emasligini tekshiruvchi kamida bitta Pest testi CI’da o‘tadi.
- ✓ Alohida migratsiyada kamida 3 ta indeks qo‘shilgan, har biri uchun `EXPLAIN ANALYZE` oldin/keyin hisobotda bor.
- ✓ Kamida bitta hisobot Redis’da teglar bilan keshlanadi; kalitda barcha parametrlar; hodisa bo‘yicha invalidation ishlaydi.
- ✓ JSON log kanali va `request_id` ishlaydi; loglarda parol, token va karta raqami yo‘q.
- ✓ `/up` va `/health` ishlaydi: baza to‘xtatilganda mos ravishda 500 va 503 qaytaradi.
- ✓ `docs/performance.md` — o‘lchov jadvali, rejalar, kesh strategiyasi va qolgan muammolar bilan.

📤 GitHub’dagi `perf` branch’idan `main` ga Pull Request havolasini topshiring: PR tavsifida `docs/performance.md` ga havola, Debugbar skrinshotlari (oldin/keyin) va `/health` javobi bo‘lsin.

Telefon kamerasi bilan skanerlang

Loyiha havolasini amaliy sahifasidagi «Loyiha bosqichi» bo‘limida topshiring.

AMALIY TOPSHIRIQ

Kompyuterda mustaqil · Oson · 10 ball

4-topshiriq: Apdex: foydalanuvchi qoniqishi

Javob vaqtlari `$ms` (millisekund) va maqsad chegarasi `$t` berilgan. Apdex = (qoniqarli + chidasa bo‘ladigan / 2) / jami. Qoniqarli: `vaqt ≤ T`; chidasa bo‘ladigan: `T < vaqt ≤ 4T`; qolganlari — norozi. Natija `round(..., 2)`. Massiv bo‘sh yoki `$t ≤ 0` bo‘lsa — `InvalidArgumentException`.

Imzo: `function apdex(array $ms, int $t): float`

Qaytaradi: `float` — 0 dan 1 gacha, 2 xona aniqlikda.

MISOL · CHAQIRUV	MISOL · NATIJA
`apdex([100, 200, 300, 1500, 2500], 500)`	`0.7`

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. PHP kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

AMALIY TOPSHIRIQ

Kompyuterda mustaqil · O‘rta · 20 ball

5-topshiriq: Strukturali log yozuvi va niqoblash

Log darajasi `$daraja` (katta-kichik harf farqsiz; ruxsat etilgan: debug, info, notice, warning, error, critical, alert, emergency), xabar `$xabar` va kontekst `$kontekst` berilgan. Kontekstni rekursiv niqoblang: kaliti (katta-kichik harf farqsiz) `password`, `token`, `api_key`, `card_number`, `secret` bo‘lgan qiymat (massiv bo‘lsa ham) `***` ga almashtiriladi; kaliti `phone` bo‘lgan satr 4 belgidan uzun bo‘lsa, oxirgi 4 belgidan boshqasi `*` bilan almashtiriladi; ichki massivlar ham shu qoidalar bilan tekshiriladi. Noma’lum daraja yoki bo‘sh (faqat bo‘sh joylardan iborat) xabar — `InvalidArgumentException`.

Imzo: `function log_yozuvi(string $daraja, string $xabar, array $kontekst): array`

Qaytaradi: `['level' => KATTA_HARFDA, 'message' => string, 'context' => array]` — kalitlar shu tartibda, kontekst kalitlari tartibi o‘zgarmaydi.

MISOL · CHAQIRUV

```
log_yozuvi('warning', 'payment.failed', ['invoice_id' => 45, 'token' => 'tok_1'])
```

MISOL · NATIJA

```
{"level":"WARNING","message":"payment.failed","context":{"invoice_id":45,"token":"***"}}
```

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. PHP kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

AMALIY TOPSHIRIQ

Kompyuterda mustaqil · O‘rta · 20 ball

6-topshiriq: Health check holatini hisoblash

Tekshiruvlar ro‘yxati berilgan: har biri `['name' => string, 'ok' => bool, 'critical' => bool, 'ms' => int]`. `failed` — `ok = false` bo‘lgan tekshiruv nomlari, `slow` — `ok = true`, lekin `ms > $sekinMs` bo‘lganlar (ikkalasi kirish tartibida). Holat: kamida bitta muhim (`critical`) tekshiruv muvaffaqiyatsiz bo‘lsa — `fail`; aks holda `failed` yoki `slow` bo‘sh bo‘lmasa — `degraded`; qolgan holatda — `ok`. HTTP kod: `fail` → 503, boshqalari → 200. Ro‘yxat bo‘sh bo‘lsa — `DomainException`.

Imzo: `function health_holati(array $tekshiruvlar, int $sekinMs = 1000): array`

Qaytaradi: `['status' => string, 'http' => int, 'failed' => array, 'slow' => array]` — kalitlar shu tartibda.

MISOL · CHAQIRUV

```
health_holati([['name' => 'db', 'ok' => true, 'critical' =>
true, 'ms' => 4], ['name' => 'queue', 'ok' => false, 'critical'
=> false, 'ms' => 2]])
```

MISOL · NATIJA

```
{"status":"degraded","http":200,"failed":["queue"],"slow":[]}
```

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. PHP kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

AMALIY TOPSHIRIQ

Kompyuterda mustaqil · Qiyin · 30 ball

7-topshiriq: Teglar va TTL bilan kesh

Redis keshining soddalashtirilgan modelini yozing. `put($key, $value, $ttl, $tags, $now)` — qiymatni `$now + $ttl` gacha saqlaydi (`$ttl ≤ 0` — `InvalidArgumentException`); `get($key, $now)` — qiymat yoki muddati o‘tgan/yo‘q bo‘lsa `null` (`$now ≥ muddat` — eskirgan); `remember($key, $ttl, $tags, $now, $fn)` — yaroqli qiymat bo‘lsa uni qaytaradi (hit), aks holda `$fn()` ni chaqirib saqlaydi (miss); `flushTag($tag)` — shu tegli barcha saqlangan yozuvlarni (muddatidan qat’i nazar) o‘chiradi va ularning sonini qaytaradi; `stats()` — faqat `remember` chaqiruvlari bo‘yicha hisob.

Imzo: `final class TagCache`

Qaytaradi: `get`/`remember` — saqlangan qiymat; `flushTag` — `int`; `stats()` — `['hits' => int, 'misses' => int]`.

MISOL · CHAQIRUV

```
(function () { $c = new TagCache(); $f = fn () => 2480000; $a = $c->remember('r:10', 600, ['reports'], 0, $f); $b = $c->remember('r:10', 600, ['reports'], 100, $f); return [$a, $b, $c->stats()]; })()
```

MISOL · NATIJA

```
[2480000,2480000,{"hits":1,"misses":1}]
```

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. PHP kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

AMALIY TOPSHIRIQ

Kompyuterda mustaqil · Qiyin · 30 ball

8-topshiriq: Takroriy so‘rovlarni aniqlash (N+1 detektori)

So‘rovlar logi `$sorovlar` (SQL satrlar) berilgan. Har bir so‘rovni normallashtiring: 1) bir tirnoqli satr literallari (ichidagi `''` bilan) → `?`; 2) alohida so‘z bo‘lgan sonlar (butun yoki o‘nlik, masalan `5`, `12.50`) → `?`; 3) `in (?, ?, ...)` ro‘yxati (katta-kichik harf farqsiz `in`) → `in (?)`; 4) ketma-ket bo‘sh joylar bitta probelga, chetlari kesiladi. Bir xil normallashgan so‘rovlarni sanang va soni `$chegara` dan kichik bo‘lmaganlarini qaytaring: soni bo‘yicha kamayish, teng bo‘lsa SQL bo‘yicha alifbo tartibida. `$chegara < 2` — `InvalidArgumentException`.

Imzo: `function takroriy_sorovlar(array $sorovlar, int $chegara): array`

Qaytaradi: `[['sql' => string, 'count' => int], ...]` — bo‘sh bo‘lishi mumkin.

MISOL · CHAQIRUV

```
takroriy_sorovlar(['select * from orders limit 50', 'select *
from counterparties where id = 3', 'select * from counterparties
where id = 7', 'select * from counterparties where id = 12'], 3)
```

MISOL · NATIJA

```
[{"sql":"select * from counterparties where id = ?","count":3}]
```

Telefon kamerasi bilan skanerlang

1. Saytga kiring (login va parol)
2. PHP kodni yozing va sinab ko‘ring
3. ✓ Topshirish — natija o‘qituvchiga

TOPSHIRIQLAR VA TOPSHIRISH

13-amaliy: 8 ta PHP topshiriq

1 **N+1: nechta SQL so‘rov?**
Oson · 10 ball · ma’ruzada

2 **Kompozit indeksning chap prefiksi**
O‘rta · 20 ball · ma’ruzada

3 **Hisobot uchun kesh kaliti**
Oson · 10 ball · ma’ruzada

4 **Apdex: foydalanuvchi qoniqishi**
Oson · 10 ball

5 **Strukturali log yozuvi va niqoblash**
O‘rta · 20 ball

6 **Health check holatini hisoblash**
O‘rta · 20 ball

7 **Teglar va TTL bilan kesh**
Qiyin · 30 ball

8 **Takroriy so‘rovlarni aniqlash (N+1 detektori)**
Qiyin · 30 ball

Telefon kamerasi bilan skanerlang

PHP kod saytda yoziladi va server tomonidan avtomatik tekshiriladi.

Loyiha havolasini ham shu sahifada topshirasiz; o‘qituvchi baho qo‘yadi.