12

O‘N IKKINCHI MA’RUZA

Korxona ilovalarini dasturlash · 12-mavzu

Korxona ilovalarini testlash

Test strategiyasi va Pest · Unit testlar · Integratsion testlar · API testlar · Qamrov va linterlar · Code review

Pest | Factories | Coverage | Larastan | Code review

ISINISH

Dasturchi QQS ni yaxlitlash qatorini “chiroyliroq” qilib qayta yozdi. Hammasi ishlayapti. Uch oydan keyin hisob-fakturalardagi QQS soliq hisobotidan 1 so‘mga farq qila boshladi. Buni kim, qachon sezishi kerak edi?

Xato kodda emas, jarayonda. Qaysi bosqich uni ushlab qolishi mumkin edi?

1

Bu o‘zgarishni qo‘lda qanday tekshirgan bo‘lardingiz va bunga qancha vaqt ketardi?

2

Kod uslubi va tur xatolarini inson ko‘rib chiqishi kerakmi yoki mashina?

3

Hamkasbingiz o‘zgarishni ko‘rib chiqqanida nimaga e’tibor bergan bo‘lardi?

O‘QUV MAQSADLARI

Bu ma’ruza oxirida siz quyidagilarni bajara olasiz…

1. **Tushuntirish** — test piramidasini: unit, integratsion, API va E2E testlarning o‘rni va narxini
2. **Yozish** — Pest da unit testlarni: expect, dataset, istisno va chegaraviy holatlar bilan
3. **Qo‘llash** — fabrikalar, RefreshDatabase va fake larni (Queue, Http, Notification) integratsion testlarda
4. **Tekshirish** — REST API ni: status kodlar, JSON tuzilmasi, validatsiya va rollar bo‘yicha huquqlar
5. **Baholash** — qamrov (coverage) ko‘rsatkichini va uning chegaralarini
6. **Sozlash** — Pint, Larastan va Pest arch testlarini hamda pre-commit hookni
7. **Tashkil etish** — Pull Request orqali kod tahlilini: tekshiruv ro‘yxati, izohlar madaniyati, branch himoyasi

01

Testlash strategiyasi va Pest asoslari

Nima uchun test · test piramidasi · Pest o‘rnatish · AAA · yaxshi test belgilari

Test piramidasi | Pest | AAA | Regressiya

TEST PIRAMIDASI

Test turlari va ularning narxi

Tur	Nima tekshiradi	Tezlik / soni	OmborPro misoli
Unit	Sof biznes-qoida, bazasiz	ms / yuzlab	`OrderStatus` o‘tishlari, QQS
Integratsion	Servis + PostgreSQL + tranzaksiya	10–100 ms / o‘nlab	`StockService::reserve()`
API (feature)	HTTP → middleware → JSON	50–300 ms / o‘nlab	`POST /api/v1/orders` → 201/422/403
E2E (brauzer)	Foydalanuvchi ssenariysi	soniyalar / bir nechta	Login → buyurtma → hisob-faktura

Pastdan yuqoriga: testlar sekinlashadi, qimmatlashadi va mo‘rtlashadi — shuning uchun soni kamayadi.

Birinchi unit test: AAA tuzilmasi

```php
// tests/Unit/VatTest.php
use App\Domain\Vat;

test('QQS 12 % bilan, so‘mgacha yaxlitlanadi', function () {
    // Arrange
    $subtotal = 1_000_005;
    // Act
    $vat = Vat::of($subtotal);
    // Assert
    expect($vat)->toBe(120_001);
});

it('nol summadan QQS nolga teng', function () {
    expect(Vat::of(0))->toBe(0);
});
```

▶ NATIJA

```
$ php artisan test --filter=Vat
   PASS  Tests\Unit\VatTest
  ✓ QQS 12 % bilan, so‘mgacha yaxlitlanadi      0.01s
  ✓ it nol summadan QQS nolga teng               0.01s
  Tests:    2 passed (2 assertions)
```

`test() / it()`

test nomi — biznes tilidagi jumla; `it` nomga “it” qo‘shadi

`Arrange / Act / Assert`

tayyorlash, bajarish, tekshirish — har doim shu tartibda

`expect(...)->toBe(...)`

qat’iy (===) taqqoslash

`1_000_005`

120 000.6 → 120 001: yaxlitlash chegarasi ataylab tanlangan

YAXSHI TEST

Yomon va yaxshi test

✗ Noto‘g‘ri

- `test('test1')` — nomidan nima tekshirilishi noma’lum
- Bitta testda 5 ta turli xatti-harakat
- `now()` va real cbu.uz ga bog‘liq — ertaga qizil
- Kutilgan qiymat production formulasidan ko‘chirilgan
- Oldingi test yaratgan yozuvga tayanadi

✓ To‘g‘ri

- `it('qoldiq yetmasa rezerv qilinmaydi')`
- Bitta xatti-harakat, aniq Arrange / Act / Assert
- `freezeTime()`, `Http::fake()` — deterministik
- Kutilgan qiymat qo‘lda hisoblangan: `120_001`
- Har test o‘z ma’lumotini fabrika bilan yaratadi

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

1-bo‘lim yakuni: Testlash strategiyasi va Pest asoslari

1

Regressiya nima va avtomatik testlar unga qanday yordam beradi?

Savol 1

2

Nima uchun piramidaning pastida testlar ko‘p, tepasida kam?

Savol 2

3

Deterministik test deganda nimani tushunasiz? Bitta buzuvchi omilni ayting.

Savol 3

MA’RUZADAGI MASHQ

Har kim o‘z qurilmasida · 10 daqiqa · Oson · 10 ball

1-topshiriq: Test natijalari xulosasi

Test yurgizuvchi (runner) natijalari ro‘yxati berilgan: har biri `['name' => string, 'status' => string, 'ms' => int]`, `status` — `passed`, `failed` yoki `skipped`. Xulosa tuzing: har bir holat soni, jami vaqt, eng sekin test nomi (bir xil bo‘lsa — ro‘yxatda birinchisi; ro‘yxat bo‘sh bo‘lsa `null`) va `ok` — yiqilgan test yo‘q **va** kamida bitta o‘tgan test bor. Noma’lum `status` — `InvalidArgumentException`.

Imzo: `function test_hisoboti(array $natijalar): array`

Qaytaradi: `['passed' => int, 'failed' => int, 'skipped' => int, 'duration_ms' => int, 'slowest' => ?string, 'ok' => bool]` — kalitlar shu tartibda.



Telefon kamerasi bilan skanerlang

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

MISOL · CHAQIRUV

```
test_hisoboti([['name' => 'qqs', 'status' => 'passed', 'ms' => 12], ['name' => 'rezerv', 'status' => 'failed', 'ms' => 80], ['name' => 'api', 'status' => 'passed', 'ms' => 150]])
```

MISOL · NATIJA

```
{"passed":2,"failed":1,"skipped":0,"duration_ms":242,"slowest":"api","ok":false}
```

02

Unit testlar: biznes-qoidalar

Sof domen kodi · expect API · dataset lar · istisnolar · chegaraviy qiymatlar

Unit | expect | Dataset | toThrow

DATASET

Holat mashinasini jadval bilan testlash

```php
use App\Enums\OrderStatus as S;

it('ruxsat etilgan o‘tishlar', function (S $from, S $to) {
    expect($from->canTransitionTo($to))->toBeTrue();
})->with([
    [S::Draft, S::Confirmed],
    [S::Confirmed, S::Reserved],
    [S::Reserved, S::Invoiced],
    [S::Invoiced, S::Shipped],
    [S::Reserved, S::Cancelled],
]);
it('taqiqlangan o‘tishlar', function (S $from, S $to) {
    expect($from->canTransitionTo($to))->toBeFalse();
})->with([
    [S::Shipped, S::Cancelled],
    [S::Invoiced, S::Cancelled],
    [S::Draft, S::Shipped],
]);
```

▶ NATIJA

```
  ✓ it ruxsat etilgan o‘tishlar with (OrderStatus::Draft,
OrderStatus::Confirmed)
  ✓ it ruxsat etilgan o‘tishlar with (OrderStatus::Confirmed,
OrderStatus::Reserved)
  ...
  ✓ it taqiqlangan o‘tishlar with (OrderStatus::Draft,
OrderStatus::Shipped)
  Tests:    8 passed (8 assertions)
```

`->with([...])`

har bir qator — alohida test, alohida natija

ruxsat va taqiq

ikkala tomonni ham tekshiring: faqat “ishlaydi” emas

`canTransitionTo()`

4-mavzudagi `OrderStatus` enumidagi o‘tish qoidasi metodi

EXPECT API

Ko‘p ishlatiladigan tekshiruvlar

Tekshiruv	Ma’nosi	Misol
`toBe($x)`	Qat’iy tenglik `===`	`expect($vat)->toBe(120_000)`
`toEqual($x)`	`==`, massivlarni chuqur	`expect($dto->toArray())->toEqual([...])`
`toMatchArray([...])`	Massivda shu kalit-qiymatlar bor	`->toMatchArray(['status' => 'draft'])`
`toHaveCount(n)`	Elementlar soni	`expect($items)->toHaveCount(3)`
`toThrow(E::class)`	Closure istisno tashlaydi	`expect(fn () => ...)->toThrow(...)`
`not->...`	Inkor	`expect($rate)->not->toBeNull()`
`->prop->toBe()`	Obyekt xossasi bo‘yicha	`expect($stock)->reserved->toBe(4)`

Decimal ustunlar Eloquent da satr qaytaradi: `toBe('1120000.00')`, `toBe(1120000)` emas.

Dataset aslida nima qiladi

```php
$allowed = ['draft>confirmed', 'draft>cancelled', 'confirmed>reserved',
    'confirmed>cancelled', 'reserved>invoiced', 'reserved>cancelled',
    'invoiced>shipped'];
$dataset = [
    ['draft', 'confirmed', true], ['reserved', 'cancelled', true],
    ['invoiced', 'cancelled', false], ['shipped', 'draft', false],
    ['draft', 'shipped', true],                // ataylab xato kutilma
];
$passed = $failed = 0;
foreach ($dataset as [$from, $to, $expected]) {
    $ok = in_array("$from>$to", $allowed, true) === $expected;
    $ok ? $passed++ : $failed++;
    echo ($ok ? 'PASS' : 'FAIL') . " $from -> $to" . PHP_EOL;
}
echo "Tests: $passed passed, $failed failed";
```

▶ NATIJA

```
PASS draft -> confirmed
PASS reserved -> cancelled
PASS invoiced -> cancelled
PASS shipped -> draft
FAIL draft -> shipped
Tests: 4 passed, 1 failed
```

`$allowed`

ruxsat etilgan o‘tishlar ro‘yxati — 4-mavzudagi qoidalar

`$dataset`

kirish + kutilgan natija jadvali

`foreach ... as [$from, $to, $expected]`

har qator — alohida tekshiruv

FAIL

kutilma noto‘g‘ri yozilgan — test ham kod, u ham xato bo‘lishi mumkin

▶ INTERAKTIV

Unit testlar

To‘g‘ri javobni tanlang. (Interaktiv slaydlarda bosib bajariladi.)

1 **Qaysi test natijasi muvaffaqiyatli?**

```
expect('1120000.00')->toBe(1120000);
```

A) O‘tadi
B) Yiqiladi: tur farq qiladi
C) Ogohlantirish beradi
D) Pest uni o‘tkazib yuboradi

2 **Istisnoni to‘g‘ri tekshiradigan yozuv qaysi?**

A) `expect($s->reserve(1, 1, 99))->toThrow(E::class)`
B) `expect(fn () => $s->reserve(1, 1, 99))->toThrow(E::class)`
C) `expect(E::class)->toThrow($s->reserve(1, 1, 99))`
D) `toThrow` faqat feature testda ishlaydi

3 **STIR validatsiyasi uchun eng foydali chegaraviy qiymatlar to‘plami?**

A) Faqat 123456789
B) 8, 9 va 10 raqamli satrlar hamda harfli satr
C) Tasodifiy 100 ta son
D) Faqat bo‘sh satr

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

2-bo‘lim yakuni: Unit testlar: biznes-qoidalar

1

Unit testga baza kerak bo‘lib qolsa, bu nimadan dalolat beradi?

Savol 1

2

`toBe` va `toEqual` farqi nimada?

Savol 2

3

Dataset holat mashinasini testlashda qanday yordam beradi?

Savol 3

MA’RUZADAGI MASHQ

Har kim o‘z qurilmasida · 10 daqiqa · O‘rta · 20 ball

2-topshiriq: Mini expect(): tekshiruvlar

Pest ning `expect()` tekshiruvlarini soddalashtirilgan ko‘rinishda yozing. `$matcher`: `toBe` (`===`), `toEqual` (`==`), `toBeTrue` (`=== true`), `toBeNull`, `toHaveCount` (massiv bo‘lsa `count === $expected`, aks holda `false`), `toContain` (massiv — qat’iy `in_array`, satr — `str_contains`, boshqa tur — `false`), `toBeGreaterThan` (`>`). `not->` prefiksi natijani inkor qiladi: `not->toBe`. Noma’lum matcher — `InvalidArgumentException`.

Imzo: `function ifoda_tekshir(mixed $actual, string $matcher, mixed $expected = null): bool`

Qaytaradi: `true` — tekshiruv o‘tdi, `false` — yiqildi.

MISOL · CHAQIRUV	MISOL · NATIJA
`ifoda_tekshir(120000, 'toBe', 120000)`	`true`

Telefon kamerasi bilan skanerlang

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

03

Integratsion testlar: baza, fabrikalar, fake lar

RefreshDatabase · test bazasi · factory, state, sequence · invariantlar · Queue/Http fake

RefreshDatabase | Factory | Http::fake | Invariant

FACTORY

ProductFactory: definition va state

```php
class ProductFactory extends Factory
{
    public function definition(): array
    {
        return [
            'sku' => fake()->unique()->bothify('UN-###'),
            'name' => fake()->words(3, true),
            'category_id' => Category::factory(),
            'unit' => fake()->randomElement(['dona', 'kg', 'quti']),
            'price' => fake()->numberBetween(1_000, 500_000) . '.00',
            'min_stock' => 10, 'is_active' => true,
        ];
    }
    public function inactive(): static
    {
        return $this->state(fn () => ['is_active' => false]);
    }
}
```

▶ NATIJA

```
Product::factory()->count(3)->create();                    // 3 ta faol
Product::factory()->inactive()->make();                    // bazaga
yozmaydi
Product::factory()->sequence(['unit' => 'kg'], ['unit' => 'dona'])
    ->count(4)->create();                                  // kg, dona,
kg, dona
```

`fake()->unique()`

`sku` unique — takrorlanmas qiymat

`Category::factory()`

bog‘liq yozuv avtomatik yaratiladi

`'.00'`

narx — decimal satr, float emas

`inactive()`

nomlangan state — testda o‘qilishi oson

INVARIANT TESTI

Qoldiq yetmasa — rezerv yo‘q, holat o‘zgarmaydi

```php
it('qoldiq yetmasa rezerv qilinmaydi', function () {
    $stock = Stock::factory()
        ->create(['quantity' => 5, 'reserved' => 3]);
    $service = app(StockService::class);

    expect(fn () => $service->reserve(
        $stock->warehouse_id, $stock->product_id, 3,
    ))->toThrow(InsufficientStockException::class);

    expect($stock->fresh())
        ->quantity->toBe(5)
        ->reserved->toBe(3);
    $this->assertDatabaseMissing('stock_movements', [
        'product_id' => $stock->product_id, 'type' => 'reserve',
    ]);
});
```

▶ NATIJA

```
  PASS  Tests\Feature\StockServiceTest
  ✓ it qoldiq yetmasa rezerv qilinmaydi          0.08s
```

`mavjud = 5 − 3 = 2`

3 ta so‘raldi — `InsufficientStockException`

`$stock->fresh()`

bazadan qayta o‘qish: holat haqiqatan o‘zgarmadimi

`assertDatabaseMissing`

jurnalga yarim yozuv tushmagan — tranzaksiya rollback

HTTP FAKE

CbuRateService: tarmoqsiz test

```php
it('CBU dan faqat kerakli valyutalarni saqlaydi', function () {
    Http::preventStrayRequests();
    Http::fake(['cbu.uz/*' => Http::response([
        ['Ccy' => 'USD', 'Rate' => '12850.45', 'Nominal' => '1',
            'Date' => '27.10.2026'],
        ['Ccy' => 'GBP', 'Rate' => '16500.00', 'Nominal' => '1',
            'Date' => '27.10.2026'],
    ])]);

    expect(app(CbuRateService::class)->fetch())->toBe(1);

    $this->assertDatabaseHas('exchange_rates', [
        'date' => '2026-10-27', 'currency' => 'USD',
    ]);
    Http::assertSentCount(1);
});
```

▶ NATIJA

```
✓ it CBU dan faqat kerakli valyutalarni saqlaydi    0.05s
(haqiqiy cbu.uz ga so‘rov ketmadi)
```

`preventStrayRequests()`

fake qilinmagan har qanday so‘rov — istisno

`Http::response([...])`

API javobini qo‘lda yozamiz — deterministik

GBP

keraksiz valyuta saqlanmasligi ham tekshiriladi

`assertSentCount(1)`

keraksiz takroriy so‘rov yo‘q

FAKE LAR

Yon ta’sirlarni almashtirish

Vosita	Tekshiruv	OmborPro misoli
`Queue::fake()`	`assertPushedOn('reports', ...)`	Hisobot jobi navbatga tushdi
`Notification::fake()`	`assertSentTo($user, ...)`	Omborchiga `LowStockNotification`
`Http::fake([...])`	`assertSent(fn ($r) => ...)`	cbu.uz, SMS provayderi
`Event::fake([...])`	`assertDispatched(OrderConfirmed::class)`	Buyurtma tasdiqlandi hodisasi
`Storage::fake('local')`	`Storage::disk('local')->assertExists()`	Hisobot fayli yaratildi
`$this->freezeTime()`	`now()` o‘zgarmaydi	Kunlik kesh kaliti, scheduler

Fake amaldan OLDIN chaqiriladi; aks holda haqiqiy job/xat/so‘rov ketib bo‘ladi.

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

3-bo‘lim yakuni: Integratsion testlar: baza, fabrikalar, fake lar

1

`RefreshDatabase` testlarni bir-biridan qanday ajratadi?

Savol 1

2

Nima uchun pul va qoldiq invariantlarini SQLite emas, PostgreSQL da testlash afzal?

Savol 2

3

`Http::preventStrayRequests()` nima uchun kerak?

Savol 3

04

API (feature) testlar

HTTP so‘rov butun stek orqali · status va JSON tasdiqlari · JWT va rollar · validatsiya

getJson / postJson

assertJsonPath

actingAs

422 / 403

GET

Mahsulotlar ro‘yxati: sahifalash va tuzilma

```php
beforeEach(function () {
    $this->seed(RoleSeeder::class);
    $this->user = User::factory()->create()->assignRole('menejer');
});

it('mahsulotlarni sahifalab qaytaradi', function () {
    Product::factory()->count(25)->create();

    $this->actingAs($this->user, 'api')
        ->getJson('/api/v1/products?per_page=10')
        ->assertOk()
        ->assertJsonCount(10, 'data')
        ->assertJsonPath('meta.total', 25)
        ->assertJsonStructure(['data' => [['sku', 'name', 'price']]]);
});
```

▶ NATIJA

```
  PASS  Tests\Feature\Api\ProductApiTest
✓ it mahsulotlarni sahifalab qaytaradi        0.21s
```

beforeEach

har testdan oldin: rollar va foydalanuvchi

actingAs($user, 'api')

JWT guardi orqali autentifikatsiya

assertJsonPath('meta.total', 25)

sahifalash meta — 7-mavzudagi format

assertJsonStructure

shartnoma: maydon nomlari o‘zgarmasin

403 va 422 ni tekshirish

```php
it('bo‘sh items bilan 422 qaytaradi', function () {
    $this->actingAs($this->user, 'api')
        ->postJson('/api/v1/orders', validOrderPayload(['items' => []]))
        ->assertUnprocessable()
        ->assertJsonValidationErrors(['items']);
});

it('rol bo‘yicha huquq', function (string $role, int $code) {
    $user = User::factory()->create()->assignRole($role);

    $this->actingAs($user, 'api')
        ->postJson('/api/v1/orders', validOrderPayload())
        ->assertStatus($code);
})->with([
    ['admin', 201], ['menejer', 201],
    ['omborchi', 403], ['buxgalter', 403],
]);
```

▶ NATIJA

```
✓ it bo‘sh items bilan 422 qaytaradi
✓ it rol bo‘yicha huquq with ('admin', 201)
✓ it rol bo‘yicha huquq with ('menejer', 201)
✓ it rol bo‘yicha huquq with ('omborchi', 403)
✓ it rol bo‘yicha huquq with ('buxgalter', 403)
```

`validOrderPayload([...])`

`tests/Pest.php` dagi yordamchi: yaroqli so‘rov + o‘zgartirish

`assertJsonValidationErrors`

aynan qaysi maydon xato ekanini tekshiradi

rol × kod dataset

huquqlar matritsasi bitta testda

FAKE + HTTP

Hisobot so‘rovi: 202 va navbatdagi job

```php
it('hisobot so‘rovi jobni navbatga qo‘yadi', function () {
    Queue::fake();
    $accountant = User::factory()->create()->assignRole('buxgalter');

    $this->actingAs($accountant)
        ->postJson('/reports/sales', [
            'from' => '2026-10-01', 'to' => '2026-10-31',
        ])
        ->assertStatus(202);

    Queue::assertPushedOn('reports', GenerateSalesReport::class,
        fn ($job) => $job->from === '2026-10-01'
            && $job->user->is($accountant));
});
```

▶ NATIJA

```
✓ it hisobot so‘rovi jobni navbatga qo‘yadi   0.09s
```

`Queue::fake()`

job bajarilmaydi, faqat navbatga qo‘yilgani yoziladi

`assertPushedOn('reports', ...)`

to‘g‘ri navbat va to‘g‘ri parametrlar

`$job->user->is(...)`

model bir xil yozuv ekanini solishtirish

TASDIQLAR

HTTP javob tasdiqlari

Usul	Kod / ma’no	Qachon
`assertOk()`	200	Ro‘yxat, ko‘rish
`assertCreated()`	201	Yangi resurs yaratildi
`assertStatus(202)`	202	Asinxron qabul qilindi
`assertUnauthorized()`	401	Token yo‘q yoki yaroqsiz
`assertForbidden()`	403	Rol yoki Policy ruxsat bermaydi
`assertUnprocessable()`	422	Validatsiya xatosi
`assertJsonPath(p, v)`	Yo‘l bo‘yicha qiymat	Maydon qiymati muhim bo‘lsa
`assertJsonStructure([...])`	Maydonlar mavjud	Shartnoma himoyasi

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

4-bo‘lim yakuni: API (feature) testlar

1

Nima uchun API testlarda `post()` emas, `postJson()` ishlatiladi?

Savol 1

2

Rollar bo‘yicha huquqlarni testlashning ixcham usuli qanday?

Savol 2

3

Faqat `assertCreated()` bilan cheklanish nima uchun yetarli emas?

Savol 3

05

Qamrov (coverage) va statik tahlil

Xdebug / PCOV · --coverage --min · qamrovni talqin qilish · Pint · Larastan · arch testlar

Coverage | Pint | Larastan | Arch

QAMROV

php artisan test --coverage --min=80

```
# Docker obrazida PCOV (dev bosqichi):
#   RUN pecl install pcov && docker-php-ext-enable pcov

docker compose exec app php artisan test --coverage --min=80

# HTML hisobot: qaysi qatorlar qoplanmagan
docker compose exec app ./vendor/bin/pest --coverage-html coverage
```

▶ NATIJA

```
Services/CbuRateService ............................ 100.0 %
Services/OrderService .................. 41..58 / 86.4 %
Services/StockService ............................... 95.2 %
Http/Controllers/ReportController ...... 22..31 / 62.5 %
Total: 81.7 %
```

PCOV / Xdebug

kengaytmasiz qamrov o‘lchanmaydi

--min=80

jami 80 % dan past — chiqish kodi xato

41..58

qoplanmagan qatorlar — qayerga test yozish kerakligi

Qamrov foizi qanday hisoblanadi

```php
$files = [
    'Services/OrderService.php' => [120, 104],   // [jami, qoplangan]
    'Services/StockService.php' => [84, 80],
    'Http/Controllers/ReportController.php' => [40, 25],
    'Models/Product.php' => [0, 0],   // bajariladigan qator yo‘q
];
$all = $hit = 0;
foreach ($files as $file => [$lines, $covered]) {
    if ($lines === 0) {
        continue;
    }
    $all += $lines;
    $hit += $covered;
    printf("%-38s %5.1f %%\n", $file, $covered / $lines * 100);
}
printf("Total: %.1f %%\n", $hit / $all * 100);
echo $hit / $all >= 0.8 ? 'OK (min 80 %)' : 'FAIL (min 80 %)';
```

▶ NATIJA

```
Services/OrderService.php                86.7 %
Services/StockService.php                95.2 %
Http/Controllers/ReportController.php    62.5 %
Total: 85.7 %
OK (min 80 %)
```

`[jami, qoplangan]`

bajariladigan va testda bajarilgan qatorlar

`$hit / $all`

jami — fayllar foizlarining o‘rtachasi emas, qatorlar yig‘indisi

`62.5 %`

jami yaxshi bo‘lsa ham, alohida fayl past bo‘lishi mumkin

phpstan.neon va natija

```
# phpstan.neon
includes:
    - vendor/larastan/larastan/extension.neon

parameters:
    paths:
        - app/
    level: 6

# ishga tushirish:
#   ./vendor/bin/phpstan analyse --memory-limit=1G
# eski loyihada boshlash:
#   ./vendor/bin/phpstan analyse --generate-baseline
```

▶ NATIJA

```
 ------ ------------------------------------------------------
  Line   app/Services/InvoiceService.php
 ------ ------------------------------------------------------
  42     Method App\Services\InvoiceService::vat() should return
         string but returns float.
 [ERROR] Found 1 error
```

`extension.neon`

Laravel “sehrini” (fasad, Eloquent) PHPStan ga tushuntiradi

`level: 6`

0–9: daraja oshgani sari qat’iyroq

`--generate-baseline`

eski xatolarni muzlatib, yangi kodni toza saqlash

ARCH TESTLAR

Arxitektura qoidalari — test ko‘rinishida

```php
// tests/Feature/ArchTest.php
arch('controllerlar DB fasadini to‘g‘ridan-to‘g‘ri ishlatmaydi')
    ->expect('App\Http\Controllers')
    ->not->toUse('Illuminate\Support\Facades\DB');

arch('debug funksiyalari qolib ketmagan')
    ->expect(['dd', 'dump', 'ray'])
    ->not->toBeUsed();

arch('modellar Eloquent Model dan meros oladi')
    ->expect('App\Models')
    ->toExtend('Illuminate\Database\Eloquent\Model');
```

▶ NATIJA

```
  ✓ arch controllerlar DB fasadini to‘g‘ridan-to‘g‘ri ishlatmaydi
  ✗ arch debug funksiyalari qolib ketmagan
    Expecting 'dd' not to be used in 'App\Services\ReportService'
(line 57).
```

arch()

Pest 2 arxitektura testlari — kodni ishga tushirmay tahlil

not->toUse(DB)

SQL faqat servis/repository qatlamida (2-mavzu)

toBeUsed

unutilgan `dd()` production ga chiqmasin

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

5-bo‘lim yakuni: Qamrov (coverage) va statik tahlil

1

Qamrov 100 % bo‘lsa, kod xatosiz degani-mi? Nima uchun?

Savol 1

2

`pint` va `pint --test` farqi nimada va qaysi biri CI da ishlatiladi?

Savol 2

3

Larastan test topa olmaydigan qanday xatoni topadi?

Savol 3

MA’RUZADAGI MASHQ

Har kim o‘z qurilmasida · 10 daqiqa · O‘rta · 20 ball

3-topshiriq: Qamrov hisoboti

Har bir fayl uchun `[jami_bajariladigan_qatorlar, qoplangan_qatorlar]` berilgan. Bajariladigan qatori 0 bo‘lgan fayllar hisobga olinmaydi. Har bir fayl foizi va **jami** foiz (qatorlar yig‘indisi bo‘yicha, fayl foizlarining o‘rtachasi emas) `round(..., 1)` bilan. `files` — foiz bo‘yicha o‘sish tartibida, teng bo‘lsa yo‘l bo‘yicha. `below` — foizi `$min` dan past fayllar (shu tartibda). `pass` — jami ≥ `$min`. `below` va `pass` **yaxlitlangan** foizlar bilan solishtiriladi. Qoplangan > jami yoki manfiy qiymat — `InvalidArgumentException`; hisobga olinadigan fayl yo‘q — `DomainException`.

Imzo: `function qamrov_hisoboti(array $fayllar, float $min): array`

Qaytaradi: `['total' => float, 'files' => [yo‘l => float, ...], 'below' => list<string>, 'pass' => bool]`.

MISOL · CHAQIRUV

```
qamrov_hisoboti(['Services/OrderService.php' => [120, 104],
'Services/StockService.php' => [84, 80],
'Http/ReportController.php' => [40, 25], 'Models/Product.php' =>
[0, 0]], 80)
```

MISOL · NATIJA

```
{"total":85.7,"files":{"Http/ReportController.php":62.5,"Service
s/OrderService.php":86.7,"Services/StockService.php":95.2},"belo
w":["Http/ReportController.php"],"pass":true}
```

Telefon kamerasi bilan skanerlang

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

06

Kod tahlili (code review) va pre-commit hooklar

Pull Request jarayoni · tekshiruv ro‘yxati · izohlar madaniyati · branch himoyasi · git hook

Pull Request | Code review | pre-commit | Branch protection

PULL REQUEST

O‘zgarish main ga qanday yetib boradi

1

Tarmoq

`git switch -c feature/low-stock-alert` — issue raqami bilan

2

Kichik commitlar

pre-commit hook: Pint, Larastan, unit testlar

3

PR ochish

Shablon: nima, nima uchun, qanday tekshirildi, skrinshotlar

4

Avtomatik tekshiruvlar

Pint, Larastan, Pest + qamrov — hammasi yashil

5

Kod tahlili

Ko‘rib chiquvchi izohlari → tuzatishlar → approve

6

Squash merge

main ga bitta toza commit; tarmoq o‘chiriladi

Branch himoyasi: to‘g‘ridan-to‘g‘ri push taqiqlangan, ≥ 1 tasdiq va yashil tekshiruvlar majburiy.

PRE-COMMIT

.githooks/pre-commit

```
#!/bin/sh
# o‘rnatish: git config core.hooksPath .githooks
FILES=$(git diff --cached --name-only --diff-filter=ACMR -- '*.php')
[ -z "$FILES" ] && exit 0

echo "Pint..."
./vendor/bin/pint --test $FILES || {
    echo "Uslub xatosi: ./vendor/bin/pint ni ishga tushiring"
    exit 1
}
echo "Larastan..."
./vendor/bin/phpstan analyse --no-progress $FILES || exit 1

echo "Unit testlar..."
./vendor/bin/pest --testsuite=Unit --stop-on-failure || exit 1
```

▶ NATIJA

```
$ git commit -m "Kam qoldiq xabari"
Pint...
  × app/Jobs/SendLowStockAlert.php   no_unused_imports
Uslub xatosi: ./vendor/bin/pint ni ishga tushiring
(commit bekor qilindi)
```

`--diff-filter=ACMR`

qo‘shilgan, nusxalangan, o‘zgargan, nomi o‘zgargan — o‘chirilganlar emas

`pint --test $FILES`

faqat staged fayllar — tez

`--testsuite=Unit`

hook da faqat tez testlar; to‘liq to‘plam — CI da

`exit 1`

noldan farqli kod — commit bekor qilinadi

TEKSHIRUV RO‘YXATI

Ko‘rib chiquvchi nimaga qaraydi

1 Biznes-qoidalar

Holat o‘tishlari, manfiy qoldiq, QQS va yaxlitlash to‘g‘rimi?

2 Tranzaksiya

Bir nechta yozuv — `DB::transaction`; qoldiq — `lockForUpdate`; job — `afterCommit`.

3 Xavfsizlik

Policy, validatsiya, `$fillable`, xom SQL da bog‘langan parametrlar, kalitlar `.env` da.

4 Unumdorlik

N+1 yo‘qmi, kerakli indeks bormi, katta ro‘yxat sahifalanganmi?

5 Testlar

Yangi qoida uchun test bormi? U xatti-harakatni tekshiradimi?

6 Migratsiya

`down()` bormi, katta jadvalda xavfsizmi, default qiymatlar?

Uslub va import tartibi — ro‘yxatda yo‘q: ularni Pint hal qiladi.

▶ INTERAKTIV

Xatoni kim ushlaydi?

Har bir muammoni uni eng ishonchli topadigan himoya qatlamiga joylashtiring. (Interaktiv slaydlarda bosib bajariladi.)

- Ishlatilmagan `use` qatori
- Metod `string` o‘rniga `float` qaytaradi
- Controllerda qolib ketgan `dd($order)`
- `shipped → cancelled` o‘tishiga ruxsat berilgan
- QQS yaxlitlashi 1 so‘mga xato
- Omborchi buyurtma yarata olyapti
- Ikki yozuv tranzaksiyasiz saqlanmoqda
- Klass nomi vazifasini aks ettirmaydi
- Yangi ustunga indeks qo‘yilmagan

Linter / statik tahlil	Avtomatik test	Code review (inson)

TUSHUNISHNI TEKSHIRISH

O‘ylang – Juftlikda – Bo‘lishing · 3 daqiqa

6-bo‘lim yakuni: Kod tahlili (code review) va pre-commit hooklar

1

Branch himoyasida qaysi uchta sozlama eng muhim?

Savol 1

2

`nit:` va `blocking:` izohlarining farqi nimada?

Savol 2

3

Pre-commit hook bo‘lsa ham nima uchun CI kerak?

Savol 3

XULOSA

Asosiy xulosalar

1 Test — bajariladigan spetsifikatsiya

Regressiyadan himoya qiladi va refaktoringni xavfsiz qiladi; piramida: ko‘p unit, kam E2E.

2 Unit — sof biznes-qoidalar

`expect`, dataset, `toThrow` (closure bilan), chegaraviy qiymatlar.

3 Integratsion — haqiqiy baza

`RefreshDatabase`, PostgreSQL test bazasi, fabrikalar, invariantlar va fake lar.

4 API test — shartnoma himoyasi

Status, JSON yo‘llari, validatsiya, rollar matritsasi va yon ta’sirlar.

5 Qamrov va linterlar

`--coverage --min`, Pint `--test`, Larastan darajalari, arch testlar.

6 Code review + hook

PR shabloni, tekshiruv ro‘yxati, izoh madaniyati; pre-commit tez, CI majburiy.

TAKRORLASH

Yakunlashdan oldin sakkizta savol

1. Test piramidasining pastki qavati?
2. Istisnoni Pest da qanday tekshiramiz?
3. `RefreshDatabase` nima qiladi?
4. Factory dagi state nima?
5. Validatsiya xatosi qaysi tasdiq bilan tekshiriladi?
6. Qamrov o‘lchash uchun nima kerak?
7. Larastan nimani tekshiradi?
8. Hook ni Git da qanday ulashamiz?

AMALIY MASHG‘ULOT

12-amaliy mashg‘ulot: 5 ta PHP topshiriq va loyiha bosqichi

4 **Factory sequence modeli**
Oson · 10 ball

5 **assertJsonPath modeli**
O‘rta · 20 ball

6 **Pre-commit: tekshiriladigan PHP fayllar**
O‘rta · 20 ball

7 **Pull Request merge ga tayyormi?**
Qiyin · 30 ball

8 **Ombor daftari: testlanadigan biznes-qoidalar**
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.

CHIQISH CHIPTASI

Darsni tark etishdan oldin

Javoblaringizni qog‘ozga yozing va o‘qituvchiga topshiring.

1

Loyihangizdagi eng xavfli biznes-qoida qaysi va uni qaysi turdagi test bilan himoyalaysiz?

2

Qamrov 85 %, lekin production da xato chiqdi. Bunga qanday izoh berasiz?

3

Bugun hamkasbingiz PR iga yozadigan birinchi `blocking:` izoh qanday bo‘lardi?