8

SAKKIZINCHI MA’RUZA

Korxona ilovalarini dasturlash · 8-mavzu

Validatsiya, import va eksport

Form Request · DB cheklovlari · Laravel Excel import · chunking va hisobot · fayllar · oraliq nazorat

Validatsiya | Yaxlitlik | Excel/CSV | Storage | Oraliq nazorat

ISINISH

Ta’minot bo‘limi 3 000 qatorli Excel fayl yubordi: “mana yangi narxlar”. Uni OmborPro ga qanday kiritasiz?

Hozircha javob bermang. Faylda nimalar noto‘g‘ri bo‘lishi mumkinligini o‘ylab ko‘ring.

1

Agar 1 750-qatorda narx o‘rniga “kelishilgan” deb yozilgan bo‘lsa, butun import to‘xtashi kerakmi?

2

Bir xil SKU faylda ikki marta uchrasa, qaysi biri to‘g‘ri hisoblanadi?

3

Validatsiya “SKU band emas” dedi, lekin bazaga yozishda xato chiqdi — bu qanday bo‘lishi mumkin?

O‘QUV MAQSADLARI

Bu ma’ruza oxirida siz quyidagilarni bajara olasiz…

1. **Qo‘llash** — Form Request: `prepareForValidation`, `rules`, `after`, maxsus qoida (`ValidationRule`)
2. **Farqlash** — ilova darajasidagi validatsiya va bazadagi cheklovlar (UNIQUE, FK, CHECK) vazifalarini
3. **Yozish** — `maatwebsite/excel` bilan mahsulotlar va kontragentlar importini (`WithHeadingRow`, `WithValidation`, `SkipsOnFailure`)
4. **Tushuntirish** — katta fayllarni bo‘laklab o‘qish (chunk) va yozish nima uchun kerakligini
5. **Loyihalash** — validatsiya hisoboti: `import_batches` jadvali va xatolar ro‘yxati
6. **Yaratish** — Excel/CSV eksport (`FromQuery`, `WithHeadings`, `WithMapping`)
7. **Qo‘llash** — Storage facade: fayl saqlash, hujjatga biriktirish, ruxsat bilan yuklab berish
8. **Tayyorlash** — oraliq nazorat uchun jamoa loyihasining taqdimotini

01

Validatsiya: Form Request va qoidalar

prepareForValidation · rules · Rule::unique · maxsus qoida · after · validated()

Form Request | rules() | ValidationRule | 422

FORM REQUEST

StoreProductRequest: normallash va qoidalar

```php
// class StoreProductRequest extends FormRequest
protected function prepareForValidation(): void
{
    $this->merge(['sku' => strtoupper(trim((string) $this->sku))]);
}

public function rules(): array
{
    return [
        'sku' => ['required', 'regex:/^[A-Z]{2,4}-\d{3,6}$/',
            Rule::unique('products')->ignore($this->product)],
        'name' => ['required', 'string', 'max:255'],
        'category_id' => ['required', 'exists:categories,id'],
        'unit' => ['required', Rule::in(Product::UNITS)],
        'price' => ['required', 'numeric', 'decimal:0,2', 'min:0'],
        'min_stock' => ['nullable', 'integer', 'min:0'],
    ];
}
```

▶ NATIJA

```
POST /products  sku=" un-001 ", price="12.345"
→ sku "UN-001" ga normallandi
→ 302 back, errors: price — “2 ta o‘nli xonadan oshmasin”
(JSON so‘rovda: 422 {"errors": {"price": [...]}})
```

`prepareForValidation()`

qoidalardan OLDIN: bo‘sh joy va kichik harf muammosi yo‘qoladi

`regex:/.../` massivda

satr ko‘rinishida | bo‘lsa qoidalar bo‘linib ketadi

`Rule::unique('products')`

ustun nomi maydon nomidan olinadi (`sku`); `ignore($this->product)` — update da o‘zi bilan to‘qnashmaydi

`Product::UNITS`

`['dona', 'kg', 'litr', 'quti']` — modeldagi konstanta, bitta manba

`decimal:0,2`

pul ko‘pi bilan 2 xona; `numeric` bilan birga `min:0` son sifatida tekshiriladi

MAXSUS QOIDA

STIR (INN) qoidasi — ValidationRule

```php
namespace App\Rules;

use Closure;
use Illuminate\Contracts\Validation\ValidationRule;

class Inn implements ValidationRule
{
    public function validate(
        string $attribute, mixed $value, Closure $fail
    ): void {
        if (! is_string($value) || ! preg_match('/^\d{9}$/', $value)) {
            $fail(':attribute 9 ta raqamdan iborat bo‘lishi kerak.');
        }
    }
}

// 'inn' => ['required', new Inn(),
//     Rule::unique('counterparties', 'inn')],
```

▶ NATIJA

```
$ php artisan make:rule Inn
INFO  Rule [app/Rules/Inn.php] created successfully.

inn = "30512345" → “STIR 9 ta raqamdan iborat bo‘lishi kerak.”
```

`make:rule Inn`

`app/Rules/Inn.php` — `ValidationRule` shabloni

`$fail(...)`

xato xabari; `:attribute` o‘rniga maydon nomi qo‘yiladi

`is_string($value)`

massiv yoki son yuborilsa ham xavfsiz

`new Inn()`

qoida obyekti boshqa qoidalar bilan bir massivda

Normallash, keyin tekshirish

```php
function normalize(array $r): array
{
    $r['sku'] = strtoupper(strtr(trim($r['sku'] ?? ''), ' _', '--'));
    $r['name'] = trim(preg_replace('/\s+/', ' ', $r['name'] ?? ''));
    return $r;
}
function errors(array $r): string
{
    $money = '/^\d+(\.\d{1,2})?$/';
    $e = [];
    if (!preg_match('/^[A-Z]{2,4}-\d{3,6}$/', $r['sku'])) $e[] = 'sku';
    if ($r['name'] === '') $e[] = 'name';
    if (!preg_match($money, $r['price'] ?? '')) $e[] = 'price';
    return '[' . implode(',', $e) . ']';
}
$a = normalize(['sku' => 'un 001', 'name' => 'Un  1', 'price' => '9']);
echo json_encode($a), ' → ', errors($a), PHP_EOL;
echo errors(normalize(['sku' => 'UN1', 'price' => '-5'])), PHP_EOL;
```

▶ NATIJA

```
{"sku":"UN-001","name":"Un 1","price":"9"} → []
[sku,name,price]
```

`strtr(..., ' _', '--')`

bo‘sh joy va `_` → `-`: `un 001` → `UN-001`

`preg_replace('/\s+/', ' ')`

nom ichidagi ortiqcha bo‘sh joylar bitta bo‘ladi

`'/^\d+(\.\d{1,2})?$/'`

narx: manfiy emas, ko‘pi bilan 2 xona; `-5` o‘tmaydi

Tartib

avval normallash, keyin tekshirish — xuddi `prepareForValidation` kabi

QOIDALAR

OmborPro maydonlari uchun qoidalar

Maydon	Qoidalar	Izoh
sku	`required`, `regex`, `Rule::unique()->ignore()`	normallash `prepareForValidation` da
price, **total**	`numeric`, `decimal:0,2`, `min:0`	pul — float emas, decimal
inn	`required`, `new Inn()`, `unique`	9 raqam, kontragentlar orasida yagona
type	`Rule::in(['customer', 'supplier', 'both'])`	kontragent turi
phone	`required_if:type,customer`, `regex:/^\+998\d{9}$/`	mijoz uchun majburiy
items.*.product_id	`required`, `distinct`, `exists:products,id`	buyurtma qatorlari
due_at	`date`, `after_or_equal:issued_at`	to‘lov muddati
file	`file`, `mimes:xlsx,csv`, `max:10240`	`max` — kilobaytda

`required` + `nullable` farqi: `nullable` — maydon bo‘lishi mumkin, lekin `null` ga ruxsat; `sometimes` — faqat kelganda tekshir.

TUSHUNISHNI TEKSHIRISH

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

1-bo‘lim yakuni: Validatsiya: Form Request va qoidalar

1

`prepareForValidation()` qachon bajariladi va unda nima qilinadi?

Savol 1

2

Nima uchun `regex` qoidasi bor bo‘lsa, qoidalarni massiv ko‘rinishida yozish kerak?

Savol 2

3

Rezerv uchun qoldiq yetarliligini Form Request da tekshirish nima uchun yetarli emas?

Savol 3

MA’RUZADAGI MASHQ

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

1-topshiriq: SKU ni normallashtirish

Foydalanuvchi SKU ni turlicha yozadi: `un_001` , `sh 0105`, `UN--001`. `prepareForValidation` mantiqini yozing: chetdagi bo‘sh joylarni olib tashlang, katta harfga o‘tkazing, bo‘sh joy va `_` belgilarini `-` ga almashtiring, ketma-ket kelgan bir nechta `-` ni bitta qiling. Natija `^[A-Z]{2,4}-\d{3,6}$` andozasiga (2–4 lotin harfi, chiziqcha, 3–6 raqam) mos kelmasa — `InvalidArgumentException`.

Imzo: `function sku_normallash(string $sku): string`

Qaytaradi: Normallangan SKU satri, masalan `UN-001`.

MISOL · CHAQIRUV	MISOL · NATIJA
`sku_normallash(' un_001 ')`	`"UN-001"`

Telefon kamerasi bilan skanerlang

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

02

Ma’lumotlar yaxlitligi: bazadagi cheklovlar

Uch qavatli himoya · UNIQUE · FOREIGN KEY · CHECK · poyga holati · UniqueConstraintViolationException

UNIQUE

FOREIGN KEY

CHECK

Poyga holati

UCH QAVATLI HIMOYA

Kim nimani ushlaydi?

1 Form Request

Shakl va format: majburiy maydonlar, SKU andozasi, `decimal:0,2`, `exists`. Foydalanuvchiga chiroyli xabar.

2 Servis qatlami

Holat va invariant: qoldiq yetarlimi, holat o‘tishi ruxsat etilganmi, to‘lov qoldiqdan oshmaydimi.

3 Baza

Oxirgi chiziq: `UNIQUE`, `FOREIGN KEY`, `CHECK`, `NOT NULL`. Har qanday yo‘l bilan kelgan ma’lumotni ushlaydi.

4 Tranzaksiya

Bir nechta yozuv — bitta butun: cheklov buzilsa, hammasi qaytariladi.

Qoidani faqat bitta qavatda yozish — xavf; lekin qavatlar bir-biriga zid bo‘lmasligi ham shart.

MIGRATSIYA

PostgreSQL CHECK cheklovlari

```php
return new class extends Migration
{
    public function up(): void
    {
        DB::statement('ALTER TABLE stocks ADD CONSTRAINT chk_stock
            CHECK (reserved >= 0 AND quantity >= reserved)');
        DB::statement('ALTER TABLE products ADD CONSTRAINT chk_price
            CHECK (price >= 0)');
        DB::statement('ALTER TABLE invoices ADD CONSTRAINT chk_paid
            CHECK (paid_amount BETWEEN 0 AND total)');
    }
    public function down(): void
    {
        DB::statement('ALTER TABLE stocks DROP CONSTRAINT chk_stock');
        DB::statement('ALTER TABLE products DROP CONSTRAINT chk_price');
        DB::statement('ALTER TABLE invoices DROP CONSTRAINT chk_paid');
    }
};
```

▶ NATIJA

```
UPDATE stocks SET reserved = 12 WHERE id = 5;   -- quantity = 10
ERROR:  new row for relation "stocks" violates check constraint
"chk_stock"
DETAIL:  Failing row contains (5, 1, 15, 10, 12).
```

`quantity >= reserved`

mavjud qoldiq = quantity − reserved ≥ 0 — invariant bazada

`DB::statement`

Laravel Blueprint da CHECK metodi yo‘q — xom SQL

`down()`

har bir cheklov qaytarib olinadigan bo‘lsin

Mavjud ma’lumot

cheklov qo‘shishdan oldin buzuvchi qatorlar tuzatiladi, aks holda migratsiya yiqiladi

POYGA HOLATI

UNIQUE buzilishini tushunarli xatoga aylantirish

```php
public function store(StoreProductRequest $request)
{
    try {
        $product = Product::create($request->validated());
    } catch (UniqueConstraintViolationException $e) {
        // validatsiyadan o‘tdi, lekin parallel so‘rov ulgurdi
        throw ValidationException::withMessages([
            'sku' => 'Bu SKU bilan mahsulot hozirgina yaratildi.',
        ]);
    }

    return redirect()->route('products.show', $product)
        ->with('status', 'Mahsulot qo‘shildi');
}
```

▶ NATIJA

So‘rov A: unique ✓ → INSERT ✓ → 302 products/51
So‘rov B: unique ✓ → INSERT ✗ (23505 unique_violation)
→ 302 back, errors.sku: “Bu SKU bilan mahsulot hozirgina yaratildi.”

UniqueConstraintViolationException

Laravel 10+ da `QueryException` ning aniq turi

`ValidationException::withMessages`

500 o‘rniga oddiy validatsiya xatosi (veb — redirect, API — 422)

23505

PostgreSQL ning unique_violation kodi

▶ INTERAKTIV

Qaysi qavat ushlaydi?

Har bir xatoni uni birinchi bo‘lib to‘xtatishi kerak bo‘lgan qavatga joylang. (Interaktiv slaydlarda bosib bajariladi.)

`price` maydoni bo‘sh yuborildi

SKU `UN001` (chiziqcha yo‘q)

STIR 8 ta raqamdan iborat

Rezerv miqdori mavjud qoldiqdan ko‘p

`shipped` buyurtmani bekor qilish

To‘lov hisob-faktura qoldig‘idan katta

Ikki parallel so‘rov bir xil SKU yaratdi

Buyurtmada ishlatilgan mahsulotni o‘chirish

Koddagi xato tufayli `reserved` manfiy bo‘ldi

Form Request

Servis qatlami

Baza cheklovi

TUSHUNISHNI TEKSHIRISH

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

2-bo‘lim yakuni: Ma’lumotlar yaxlitligi: bazadagi cheklovlar

1

Nima uchun `unique:products,sku` validatsiya qoidasi bo‘lsa ham bazada UNIQUE indeks kerak?

Savol 1

2

PostgreSQL da CHECK cheklovi Laravel migratsiyasida qanday qo‘shiladi?

Savol 2

3

`withoutTrashed()` li unique qoidasi va oddiy UNIQUE indeks birga qanday muammo tug‘diradi?

Savol 3

MA’RUZADAGI MASHQ

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

2-topshiriq: Mahsulot validatsiyasi (xatolar lug‘ati)

Form Request ga o‘xshash tekshiruvchi yozing. Kirish — maydonlar massivi (qiymatlar satr yoki son). Maydonlar shu tartibda tekshiriladi va har biriga ko‘pi bilan bitta xabar: `sku` — yo‘q/bo‘sh: “SKU majburiy”, andozaga (`^[A-Z]{2,4}-\d{3,6}$`, `trim` dan keyin) mos emas: “SKU formati noto‘g‘ri”; `name` — bo‘sh: “Nomi majburiy”, 255 baytdan uzun: “Nomi juda uzun”; `unit` — `dona`, `kg`, `litr`, `quti` dan biri emas: “Birlik noto‘g‘ri”; `price` — manfiy bo‘lmagan, ko‘pi bilan 2 o‘nli xonali son (`^\d+(\.\d{1,2})?$`) emas: “Narx noto‘g‘ri”; `min_stock` — ixtiyoriy, berilgan bo‘lsa faqat raqamlar: “Minimal qoldiq noto‘g‘ri”.

Imzo: `function mahsulot_xatolari(array $r): array`

Qaytaradi: `['maydon' => ['xabar'], ...]` — faqat xatoli maydonlar, yuqoridagi tartibda; xato yo‘q bo‘lsa `[]`.

MISOL · CHAQIRUV

```
mahsulot_xatolari(['sku' => 'UN-001', 'name' => 'Un oliy nav',
'unit' => 'dona', 'price' => '385000'])
```

MISOL · NATIJA

```
[]
```

Telefon kamerasi bilan skanerlang

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

03

Ommaviy import: maatwebsite/excel

Laravel Excel concerns · WithHeadingRow · WithValidation · SkipsOnFailure · CSV xususiyatlari

Laravel Excel | ToModel | WithValidation | CSV

CONCERNS

Import sinfiga nimani qo‘shamiz?

Concern	Nima beradi	Metod / trait
ToModel	har bir qator → Eloquent model	`model(array $row)`
ToCollection	qatorlarni o‘zingiz qayta ishlaysiz	`collection(Collection $rows)`
WithHeadingRow	sarlavhalar kalit bo‘ladi (slug)	`headingRow(): int` (ixtiyoriy)
WithValidation	har qatorga Laravel qoidalari	`rules()`, `prepareForValidation()`
SkipsOnFailure	xato qatorlar o‘tkaziladi, yig‘iladi	`SkipsFailures` → `failures()`
WithChunkReading	fayl bo‘laklab o‘qiladi	`chunkSize(): int`
WithBatchInserts	guruhlab `INSERT`	`batchSize(): int`
WithUpserts	mavjud bo‘lsa yangilash	`uniqueBy()` — `'sku'`

`SkipsOnFailure` bo‘lmasa, birinchi yaroqsiz qatorda `ValidationException` tashlanadi va import to‘xtaydi.

APP/IMPORTS

ProductsImport

```php
class ProductsImport implements ToModel, WithHeadingRow,
    WithValidation, SkipsOnFailure, WithChunkReading
{
    use Importable, SkipsFailures;
    public int $okRows = 0;

    public function model(array $row): ?Product
    {
        $this->okRows++;          // faqat validatsiyadan o‘tganlar
        return new Product([
            'sku' => strtoupper(trim($row['sku'])),
            'name' => $row['nomi'], 'unit' => $row['birlik'],
            'price' => $row['narx'],
            'category_id' => $row['kategoriya_id'],
        ]);
    }
    public function chunkSize(): int { return 1000; }
}
```

▶ NATIJA

```
Shablon: | sku | nomi | kategoriya_id | birlik | narx | min_qoldiq
|
$row = ['sku' => 'UN-001', 'nomi' => 'Un oliy nav, 50 kg',
        'kategoriya_id' => 3, 'birlik' => 'dona', 'narx' =>
385000, ...]
```

WithHeadingRow

`$row['nomi']` — ustun raqami emas, sarlavha nomi

SkipsFailures

yaroqsiz qatorlar `failures()` to‘plamiga tushadi

`model()`

faqat qoidalardan o‘tgan qatorlar uchun chaqiriladi

`chunkSize()`

fayl 1000 qatorlik bo‘laklarda o‘qiladi — xotira tejaladi

QOIDALAR

Qatorlar uchun rules va prepareForValidation

```php
public function prepareForValidation(array $data, int $index): array
{
    $data['sku'] = strtoupper(trim((string) ($data['sku'] ?? '')));
    $data['narx'] = str_replace([' ', ','], ['', '.'],
        (string) ($data['narx'] ?? ''));
    return $data;
}
public function rules(): array
{
    return [
        'sku' => ['required', 'regex:/^[A-Z]{2,4}-\d{3,6}$/',
            'unique:products,sku'],
        'nomi' => ['required', 'string', 'max:255'],
        'birlik' => ['required', 'in:dona,kg,litr,quti'],
        'narx' => ['required', 'numeric', 'min:0'],
        'kategoriya_id' => ['required', 'exists:categories,id'],
    ];
}
```

▶ NATIJA

```
Failure: row() = 7, attribute() = 'narx',
         errors() = ['The narx field must be a number.'],
         values() = ['sku' => 'SH-010', 'narx' => 'kelishilgan',
...]
```

`prepareForValidation($data, $index)`

Laravel Excel dagi qator darajasidagi normallash

`'385 000,50' → '385000.50'`

mintaqaviy son formati tozalanadi

`'unique:products,sku'`

faqat bazaga nisbatan; fayl ichidagi takror alohida

`customValidationMessages()`

o‘zbekcha xabarlar uchun qo‘shimcha metod

CSV qatorini o‘qish va normallash

```php
<?php
$csv = "sku;nomi;narx\nUN-001;Un oliy nav;385 000\n"
    . "SH-010;\"Shakar; 1 kg\";14 500,50\n";
$lines = array_values(array_filter(explode("\n", $csv)));
$head = str_getcsv(array_shift($lines), ';', '"', '');

foreach ($lines as $i => $line) {
    $row = array_combine($head, str_getcsv($line, ';', '"', ''));
    $row['narx'] = str_replace([' ', ','], ['', '.'], $row['narx']);
    printf("%d-qator: %s | %s | %s\n",
        $i + 2, $row['sku'], $row['nomi'], $row['narx']);
}
```

▶ NATIJA

```
2-qator: UN-001 | Un oliy nav | 385000
3-qator: SH-010 | Shakar; 1 kg | 14500.50
```

`str_getcsv(..., ';')`

ajratuvchi `;`; qo‘shtirnoq ichidagi `;` maydon qismi

`array_combine($head, ...)`

sarlavha → kalit, xuddi `WithHeadingRow` kabi

`$i + 2`

Excel qator raqami: sarlavha 1-qator, ma’lumot 2-dan

`'14 500,50' → '14500.50'`

bo‘sh joy va o‘nli vergul tozalandi

TUSHUNISHNI TEKSHIRISH

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

3-bo‘lim yakuni: Ommaviy import: maatwebsite/excel

1

`ToModel` va `ToCollection` concern larining farqi nimada?

Savol 1

2

`Failure` obyektidan hisobot uchun qanday ma’lumotlarni olamiz?

Savol 2

3

Excel dan saqlangan CSV ni o‘qishda qanday ikki tipik muammo uchraydi?

Savol 3

MA’RUZADAGI MASHQ

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

3-topshiriq: Sarlavhadan kalit (heading row)

`WithHeadingRow` sarlavhalarni kalitga aylantiradi. Soddalashtirilgan variantini yozing: chetdagi bo‘sh joylarni olib tashlang, o‘zbekcha tutuq va o‘ belgilari (‘, ’, ‘, ASCII `'`) ni olib tashlang, kichik harfga o‘tkazing (`strtolower`), lotin harfi va raqamdan boshqa har qanday belgilar ketma-ketligini bitta `_` ga almashtiring, boshidagi va oxiridagi `_` ni olib tashlang. Natija bo‘sh bo‘lsa — `InvalidArgumentException`.

Imzo: `function sarlavha_kalit(string $sarlavha): string`

Qaytaradi: Kalit satri, masalan `narx_som`.

MISOL · CHAQIRUV	MISOL · NATIJA
`sarlavha_kalit('Narx (so‘m)')`	`"narx_som"`

Telefon kamerasi bilan skanerlang

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

04

Chunking, validatsiya hisoboti va eksport

Xotira va vaqt · chunkById · upsert · import_batches · FromQuery eksport · CSV va BOM

Chunk | import_batches | Eksport | upsert

Xatolardan import hisobotini yig‘ish

```php
<?php
$okRows = 7;                        // model() chaqirilgan qatorlar
$failures = [
    ['row' => 4, 'field' => 'sku', 'message' => 'SKU band'],
    ['row' => 4, 'field' => 'narx', 'message' => 'Narx son emas'],
    ['row' => 9, 'field' => 'birlik', 'message' => 'Birlik noto‘g‘ri'],
];
$failed = count(array_unique(array_column($failures, 'row')));
$report = [
    'total_rows' => $okRows + $failed,
    'ok_rows' => $okRows,
    'failed_rows' => $failed,
];
echo json_encode($report), PHP_EOL;
foreach ($failures as $f) {
    printf("%d-qator, %s: %s\n", $f['row'], $f['field'], $f['message']);
}
```

▶ NATIJA

```
{"total_rows":9,"ok_rows":7,"failed_rows":2}
4-qator, sku: SKU band
4-qator, narx: Narx son emas
9-qator, birlik: Birlik noto‘g‘ri
```

`array_unique(array_column(...))`

bitta qatorda bir nechta xato — qator bir marta sanaladi

`total = ok + failed`

bo‘sh qatorlar hisobga kirmaydi

errors json

shu massiv `import_batches.errors` ga yoziladi

CONTROLLER

Import + hisobotni saqlash

```php
public function store(Request $request)
{
    $request->validate(['file' => 'required|file|mimes:xlsx,csv,txt']);
    $path = $request->file('file')->store('imports');
    $import = new ProductsImport();
    Excel::import($import, $path);
    $errors = $import->failures()->map(fn (Failure $f) => [
        'row' => $f->row(), 'field' => $f->attribute(),
        'message' => $f->errors()[0],
    ])->values();
    $failed = $errors->pluck('row')->unique()->count();
    $batch = ImportBatch::create([
        'file_path' => $path, 'status' => 'done', 'errors' => $errors,
        'ok_rows' => $import->okRows, 'failed_rows' => $failed,
        'total_rows' => $import->okRows + $failed,
        'user_id' => $request->user()->id]);
    return redirect()->route('imports.show', $batch);
}
```

▶ NATIJA

```
→ 302 /imports/17
Import #17: 2 312 / 2 340 qator yuklandi, 28 qatorda xato
[Xatolarni Excel da yuklab olish]
```

`store('imports')`

fayl tasodifiy nom bilan `local` diskka — keyin qayta tekshirish uchun

`Excel::import($import, $path)`

standart diskdagi faylni o‘qiydi

`$import->failures()`

`Failure` obyektlari to‘plami → json hisobot

`ImportBatch::create`

`errors` ustuni modelda `array` cast — json bo‘lib saqlanadi

APP/EXPORTS

ProductsExport — FromQuery

```php
class ProductsExport implements FromQuery, WithHeadings, WithMapping
{
    public function query()
    {
        return Product::query()->with('category')->orderBy('sku');
    }

    public function headings(): array
    {
        return ['SKU', 'Nomi', 'Kategoriya', 'Birlik', 'Narx (so‘m)'];
    }

    public function map($product): array
    {
        return [$product->sku, $product->name,
            $product->category?->name, $product->unit, $product->price];
    }
}
```

▶ NATIJA

```
return Excel::download(new ProductsExport(), 'mahsulotlar.xlsx');
return Excel::download(new ProductsExport(), 'mahsulotlar.csv',
    \Maatwebsite\Excel\Excel::CSV);
→ Content-Disposition: attachment; filename=mahsulotlar.xlsx
```

FromQuery

so‘rov bo‘laklab (`exports.chunk_size`) bajariladi — xotira tejaladi

`with('category')`

har bir qator uchun alohida so‘rov (N+1) bo‘lmaydi

`WithMapping::map()`

ustunlar tartibi va formati — sarlavhalar bilan bir xil

`use_bom`

CSV ni Excel to‘g‘ri ochishi uchun UTF-8 BOM

KATTA HAJM

Butun jadval va bo‘laklab ishlash

✗ Noto‘g‘ri

- `Product::all()->each(...)` — butun jadval xotirada
- `chunk()` ichida filtr ustunini o‘zgartirish — yozuvlar o‘tib ketadi
- 50 000 qatorni bitta tranzaksiyada yozish
- fayldagi takroriy SKU bilan bitta `upsert` — PostgreSQL xatosi

✓ To‘g‘ri

- `chunkById(1000, ...)`, `lazyById()`, `cursor()`
- `WithChunkReading` + `WithBatchInserts` importda
- har bir bo‘lak — alohida tranzaksiya, natija hisobotda
- takrorlarni oldindan aniqlab, hisobotga yozish

TUSHUNISHNI TEKSHIRISH

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

4-bo‘lim yakuni: Chunking, validatsiya hisoboti va eksport

1

Nima uchun filtr ustunini o‘zgartirayotganda `chunk()` emas, `chunkById()` ishlatiladi?

Savol 1

2

`import_batches` jadvalida qaysi ustunlar bor va ular nima uchun kerak?

Savol 2

3

Eksportda `FromQuery` nima uchun `FromCollection` dan afzal?

Savol 3

05

Fayllarni saqlash va hujjatlarga biriktirish

Storage facade · disklar · store va download · polimorf attachments · xavfsizlik

Storage | Disk | morphMany | Xavfsizlik

MIGRATSIYA

Polimorf attachments jadvali

```php
Schema::create('attachments', function (Blueprint $table) {
    $table->id();
    $table->morphs('attachable');   // attachable_type, attachable_id
    $table->string('disk', 20)->default('local');
    $table->string('path');
    $table->string('original_name');
    $table->string('mime', 100);
    $table->unsignedBigInteger('size');
    $table->foreignId('user_id')->constrained();
    $table->timestamps();
});
```

▶ NATIJA

```
attachable_type = 'App\Models\Order',   attachable_id = 42
path = 'attachments/orders/42/kX3f9aQ...pdf'
original_name = 'Buyurtma xati 15-oktabr.pdf'
```

`morphs('attachable')`

ikki ustun + ularga indeks

`path / original_name`

diskdagi xavfsiz nom va foydalanuvchiga ko‘rsatiladigan nom alohida

`Order::attachments()`

`return $this->morphMany(Attachment::class, 'attachable');`

`Attachment::attachable()`

`return $this->morphTo();` — qaysi hujjatga tegishli

YUKLASH

Hujjatga fayl biriktirish

```php
public function store(Request $request, Order $order)
{
    $this->authorize('update', $order);
    $request->validate(['file' => ['required', 'file',
        'mimes:pdf,jpg,png', 'extensions:pdf,jpg,png', 'max:5120']]);

    $file = $request->file('file');
    $order->attachments()->create([
        'path' => $file->store("attachments/orders/{$order->id}"),
        'original_name' => $file->getClientOriginalName(),
        'mime' => $file->getMimeType(),
        'size' => $file->getSize(),
        'user_id' => $request->user()->id,
    ]);

    return back()->with('status', 'Fayl biriktirildi');
}
```

▶ NATIJA

```
invoice.php (yoki .pdf ga qayta nomlangan PHP) → 422 “file must be
a file of type: pdf, jpg, png”
skaner.pdf (2,3 MB) → saqlandi:
attachments/orders/42/kX3f9aQ....pdf
```

`mimes + extensions`

mazmun ham, nom ham tekshiriladi

`max:5120`

kilobaytda — 5 MB

`store(...)`

tasodifiy nom, standart `local` disk — vebdan ochiq emas

`getMimeType()`

server aniqlagan tur; `getClientMimeType()` — mijoz aytgani (ishonmang)

YUKLAB BERISH

Ruxsat bilan download va o‘chirish

```php
public function download(Attachment $attachment)
{
    $this->authorize('view', $attachment->attachable);

    return Storage::disk($attachment->disk)
        ->download($attachment->path, $attachment->original_name);
}

public function destroy(Attachment $attachment)
{
    $this->authorize('update', $attachment->attachable);
    $attachment->delete();                                  // avval yozuv
    Storage::disk($attachment->disk)->delete($attachment->path);

    return back()->with('status', 'Fayl o‘chirildi');
}
```

`authorize('view', ...attachable)`

faylga ruxsat — u biriktirilgan hujjatga ruxsat orqali

`download($path, $name)`

foydalanuvchi asl nomni ko‘radi, diskda esa hash nom

avval yozuv, keyin fayl

xato bo‘lsa “yetim” fayl qoladi, lekin buzuq havola emas

▶ NATIJA

```
GET /attachments/9/download (omborchi, ruxsat yo‘q) → 403
GET /attachments/9/download (menejer) → 200
Content-Disposition: attachment; filename="Buyurtma xati 15-
oktabr.pdf"
```

TUSHUNISHNI TEKSHIRISH

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

5-bo‘lim yakuni: Fayllarni saqlash va hujjatlarga biriktirish

1

`local` va `public` disklarining farqi nimada va hujjatlar qayerda saqlanadi?

Savol 1

2

Nima uchun `getClientOriginalName()` ni saqlash yo‘li sifatida ishlatmaymiz?

Savol 2

3

Polimorf `attachments` jadvali qanday muammoni hal qiladi?

Savol 3

06

Oraliq nazorat: loyiha taqdimoti

Jamoa taqdimoti · jonli demo · repozitoriy va doska · savol-javob

Oraliq nazorat | Taqdimot | Demo

TAQDIMOT TUZILMASI

10 daqiqa: nimani va qanday ko‘rsatamiz

1

1 daqiqa

Jamoa, soha, loyiha maqsadi — xartiyadan bir slayd

2

2 daqiqa

ER-sxema va arxitektura: qatlamlar, asosiy servislar

3

3–4 daqiqa

Jonli demo: buyurtma oqimi, rollar, CRUD, API `/docs`, import

4

1 daqiqa

GitHub: doska, PR lar, har bir a’zoning hissasi

5

3 daqiqa

Savol-javob: har bir talaba o‘z qismi bo‘yicha

Demo — seed ma’lumotlarda; zaxira sifatida 2–3 daqiqalik ekran yozuvi tayyor bo‘lsin.

BAHOLASH

Nimaga qaraymiz

Mezon	Nimani ko‘ramiz	Dalil
Funksionallik	asosiy oqim ishlaydi, invariantlar buzilmaydi	jonli demo
Arxitektura va kod	qatlamlar, servis, Form Request, Resource	repozitoriy, PR
Ma’lumotlar	migratsiyalar, cheklovlar, seed	ER-sxema, `migrate:fresh --seed`
Jamoa ishi	vazifalar taqsimoti, kod tahlili	doska, commit tarixi
Taqdimot	aniq tushuntirish, savollarga javob	savol-javob

Ball taqsimoti fan dasturidagi oraliq nazorat bo‘yicha alohida e’lon qilinadi.

TUSHUNISHNI TEKSHIRISH

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

6-bo‘lim yakuni: Oraliq nazorat: loyiha taqdimoti

1

Oraliq nazorat taqdimotida jonli demoda qaysi oqim albatta ko‘rsatiladi?

Savol 1

2

Jamoa ishi qanday dalillar bilan baholanadi?

Savol 2

3

Demo ishlamay qolsa nima qilinadi?

Savol 3

XULOSA

Asosiy xulosalar

1 Validatsiya chegarada

Form Request: `prepareForValidation` → `rules` → `after`; controller faqat `validated()` ni oladi.

2 Uch qavatli himoya

Form Request — format, servis — holat, baza — UNIQUE, FK, CHECK; ular bir-biriga zid bo‘lmasin.

3 Poyga holati

Validatsiya parallel so‘rovlarni to‘xtatmaydi — UNIQUE indeks va `UniqueConstraintViolationException`.

4 Laravel Excel

`WithHeadingRow`, `WithValidation`, `SkipsOnFailure`, `WithChunkReading`; `failures()` — hisobot manbai.

5 Hisobot va eksport

`import_batches` da natija va xatolar; eksport `FromQuery` bilan bo‘laklab, CSV uchun BOM.

6 Fayllar

Storage disklari, `store()` tasodifiy nom, polimorf `attachments`, yuklab berish faqat `authorize()` bilan.

TAKRORLASH

Yakunlashdan oldin sakkizta savol

1. `prepareForValidation()` nima uchun kerak?
2. `decimal:0,2` qoidasi nimani tekshiradi?
3. Bazada CHECK qanday qo‘shiladi?
4. Poyga holatida SKU takrorini kim to‘xtatadi?
5. `SkipsOnFailure` bo‘lmasa nima bo‘ladi?
6. Nima uchun `chunkById` ?
7. `mimes` va `extensions` farqi?
8. Hujjat faylini qanday yuklab beramiz?

AMALIY MASHG‘ULOT

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

4 **Import bo‘laklari (chunk)**
Oson · 10 ball

5 **Import validatsiya hisoboti**
O‘rta · 20 ball

6 **Kontragent qatorini normallashtirish**
O‘rta · 20 ball

7 **Upsert rejasi: qo‘shish, yangilash, o‘tkazish**
Qiyin · 30 ball

8 **Biriktirma fayl uchun xavfsiz yo‘l**
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

Kontragentlar importida qaysi 3 ta xatoni Form Request/Excel qoidasi, qaysi birini baza ushlaydi?

2

Foydalanuvchi yuklagan faylni nima uchun `public` diskka va asl nomi bilan saqlamaslik kerak?

3

Oraliq nazorat demosida jamoangiz qaysi oqimni ko‘rsatadi va unga nima yetishmayapti?