2

IKKINCHI MA’RUZA

Korxona ilovalarini dasturlash · 2-mavzu

Qatlamli arxitektura va loyihalash namunalari

Qatlamli arxitektura · MVC va Laravel · Repository va Service Layer · DTO va Unit of Work · Loyiha strukturasi

Qatlamlar | MVC | Service Layer | Repository | DTO

ISINISH

Buyurtma yaratish kodi 300 qatorlik controller metodida yozilgan. Ertaga xuddi shu buyurtmani API va Excel importi orqali ham yaratish kerak. Nima qilasiz?

Kodni nusxalash yagona yo‘lmi? Avval o‘ylab ko‘ring, keyin muhokama qilamiz.

1

Kodni uch joyga nusxalasangiz, QQS qoidasi o‘zgarganda nechta joyni tuzatasiz?

2

Bu metodni ma’lumotlar bazasisiz qanday testlaysiz?

3

Qaysi qatorlar HTTP ga, qaysilari biznesga, qaysilari bazaga tegishli?

O‘QUV MAQSADLARI

Bu ma’ruza oxirida siz quyidagilarni bajara olasiz…

1. **Tushuntirish** — qatlamli arxitekturaning maqsadini, qatlamlar vazifasini va bog‘liqlik yo‘nalishini
2. **Tavsiflash** — Laravel da HTTP so‘rovning hayot siklini va MVC ning har bir qismi vazifasini
3. **Qo‘llash** — Service Layer va Repository namunalarini bog‘liqlik in’ektsiyasi (dependency injection) bilan
4. **Yozish** — o‘zgarmas (readonly) DTO sinflarini va ularni so‘rovdan yaratadigan fabrika metodlarini
5. **Farqlash** — Unit of Work namunasini va Laravel dagi `DB::transaction` yondashuvini
6. **Baholash** — qachon Repository foydali, qachon esa ortiqcha murakkablik ekanini
7. **Loyihalash** — Laravel 12 loyihasining papka strukturasini OmborPro modullari uchun

01

Qatlamli arxitektura

Vazifalarni ajratish · to‘rt qatlam · bog‘liqlik yo‘nalishi · semiz controller

Qatlamlar

Separation of concerns

Bog‘liqlik

TO‘RT QATLAM

Qatlamlar va ularning vazifasi

1

Taqdimot

Controller, Form Request, Blade, Livewire, API Resource — kirish va javob

2

Ilova (servis)

`OrderService`, `StockService` — jarayon, tranzaksiya chegarasi

3

Domen

Modellar, `OrderStatus` enumi, qiymat obyektlari, domen istisnolari

4

Ma’lumot / infratuzilma

Eloquent, `ProductRepository`, tashqi API, navbatlar, fayllar

Bog‘liqlik faqat yuqoridan pastga: pastki qatlam yuqoridagini bilmaydi.

ANTI-NAMUNA

Semiz va ingichka controller

Noto‘g‘ri: semiz controller

- Validatsiya, narx, QQS, qoldiq — bitta metodda
- API va import uchun kod nusxalanadi
- Testlash uchun HTTP so‘rov va baza kerak
- Tranzaksiya chegarasi noaniq

To‘g‘ri: ingichka controller

- Validatsiya — `StoreOrderRequest` da
- Jarayon — `OrderService::create()` da
- Controller 5–10 qator: chaqiradi va javob qaytaradi
- Servisni bazasiz, soxta repository bilan testlash mumkin

Qatlamlar soni masalaga mos bo‘lsin: kategoriyalar CRUD i uchun servis shart emas.

Uch qatlam — uch funksiya

```php
function topish(string $sku): ?array        // ma’lumot qatlami
{
    return ['UN-001' => ['price' => 15000]][$sku] ?? null;
}
function narxla(string $sku, int $qty): int  // biznes qatlami
{
    $p = topish($sku) ?? throw new DomainException("$sku topilmadi");
    return $p['price'] * $qty;
}
// taqdimot qatlami: kirish → javob (HTTP kodi)
foreach ([['UN-001', '3'], ['XX-999', '1']] as [$sku, $qty]) {
    try {
        $jami = narxla($sku, (int) $qty);
        echo "200 $jami\n";
    } catch (DomainException $e) {
        echo "422 ", $e->getMessage(), "\n";
    }
}
```

▶ NATIJA

```
200 45000
422 XX-999 topilmadi
```

topish()

qayerdan o‘qishni faqat shu funksiya biladi (massiv, PostgreSQL, API)

narxla()

qoida: mahsulot topilmasa — domen istisnosi

(int) $qty

satrni turga keltirish — taqdimot qatlamining ishi

200 / 422

istisnoni HTTP javobiga aylantirish ham taqdimotda

▶ INTERAKTIV

Bu kod qaysi qatlamga tegishli?

Har bir holat uchun to‘g‘ri qatlamni tanlang. (Interaktiv slaydlarda bosib bajariladi.)

1 **Mavjud qoldiq yetarli bo‘lmasa `InsufficientStockException` tashlash.**

A) Taqdimot
B) Servis / domen
C) Blade ko‘rinishi
D) Migratsiya

2 **`counterparty_id` maydoni majburiy va `counterparties` jadvalida mavjud ekanini tekshirish.**

A) Form Request (taqdimot)
B) Repository
C) Eloquent model
D) Navbat ishchisi

3 **Kam qoldiqli mahsulotlarni JOIN bilan tanlaydigan murakkab so‘rov.**

A) Controller
B) Blade
C) Repository (ma’lumot qatlami)
D) Middleware

TUSHUNISHNI TEKSHIRISH

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

1-bo‘lim yakuni: Qatlamli arxitektura

1

Qatlamli arxitekturada bog‘liqlik qaysi yo‘nalishda bo‘ladi?

Savol 1

2

Semiz controller ning uchta kamchiligini ayting.

Savol 2

3

Qaysi holatda servis qatlamini yozmaslik oqilona?

Savol 3

MA’RUZADAGI MASHQ

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

1-topshiriq: Qatlam qoidasini buzgan bog‘liqliklar

Sinflar va ular foydalanadigan sinflar ro‘yxati berilgan: `['OrderController' => ['OrderService'], ...]`. Qatlam sinf nomining oxiri bo‘yicha aniqlanadi: `...Controller` — 1, `...Service` — 2, `...Repository` — 3, qolganlari (modellar, enumlar) — 4. Sinf faqat o‘z qatlamidagi yoki pastroq qatlamdagi (raqami katta yoki teng) sinfga bog‘lanishi mumkin; yuqoriga qaragan bog‘liqlik — buzilish. Barcha buzilishlarni toping.

Imzo: `function qatlam_buzilishlari(array $bogliqlik): array`

Qaytaradi: `'Sinf -> Bog‘liqlik'` ko‘rinishidagi satrlar ro‘yxati, alifbo tartibida, takrorlarsiz. Buzilish yo‘q bo‘lsa — `[]`.

MISOL · CHAQIRUV

```
qatlam_buzilishlari(['OrderController' => ['OrderService'],
'OrderService' => ['ProductRepository', 'OrderController']])
```

MISOL · NATIJA

```
["OrderService -> OrderController"]
```

Telefon kamerasi bilan skanerlang

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

02

MVC va Laravel da so‘rov hayot sikli

Model · View · Controller · index.php dan javobgacha · middleware

MVC | Controller | Blade | Middleware

Model, View, Controller — kim nima qiladi

MVC taqdimot qatlamini tartiblaydi: ma’lumot, ko‘rinish va boshqaruv alohida.

Model

Eloquent modeli: `Order`, `Product` — jadval, bog‘lanishlar, turlar (casts).

View

Blade shabloni yoki JSON (API Resource) — faqat tayyor ma’lumotni ko‘rsatadi.

Controller

So‘rovni qabul qiladi, servis yoki modelni chaqiradi, javobni tanlaydi.

+ Servis

Biznes-jarayon M ga ham, C ga ham emas — alohida `OrderService` ga.

HAYOT SIKLI

So‘rov index.php dan javobgacha

1

public/index.php

Autoloader va `bootstrap/app.php` dan ilova obyekti

2

HTTP kernel

Global middleware: CSRF, sessiya, cookie shifrlash

3

Provayderlar

`AppServiceProvider` — bog‘lanishlar ro‘yxatdan o‘tadi

4

Router

Mos marshrut, marshrut middleware lari (`auth`)

5

Controller

Request validatsiyasi, servis chaqiruvi

6

Response

Middleware lar orqali teskari qaytadi va brauzerga yuboriladi

`php artisan route:list` — barcha marshrutlar, ularning controller va middleware lari.

CONTROLLER

Marshrut va ingichka controller

```php
// routes/web.php
Route::get('/orders', [OrderController::class, 'index'])
    ->name('orders.index');

// app/Http/Controllers/OrderController.php
class OrderController extends Controller
{
    public function index(): View
    {
        $orders = Order::with('counterparty')
            ->latest()
            ->paginate(20);

        return view('orders.index', ['orders' => $orders]);
    }
}
```

▶ NATIJA

```
GET /orders → 200 OK, text/html
SQL: select * from orders order by created_at desc limit 20
SQL: select * from counterparties where id in (3, 7, 12)
```

`->name('orders.index')`

nomli marshrut: `route('orders.index')` bilan URL yasaladi

`with('counterparty')`

bog‘langan kontragentlar bitta qo‘shimcha so‘rovda olinadi

`latest()`

`created_at` bo‘yicha kamayish tartibida

`view(...)`

ko‘rinish nomi va unga beriladigan ma’lumot

VIEW

Blade — faqat ko‘rsatish

```blade
@extends('layouts.app')

@section('content')
<table>
    @foreach ($orders as $order)
        <tr>
            <td>{{ $order->number }}</td>
            <td>{{ $order->counterparty->name }}</td>
            <td>{{ number_format($order->total, 2, '.', ' ') }}</td>
            <td>{{ $order->status->value }}</td>
        </tr>
    @endforeach
</table>
{{ $orders->links() }}
@endsection
```

▶ NATIJA

```
BY-2026-000123 | Baraka savdo MChJ | 1 120 000.00 | confirmed
BY-2026-000122 | Nur market       |   450 000.00 | draft
« 1 2 3 … 7 »
```

`{{ ... }}`

chiqish avtomatik ekranlanadi (XSS dan himoya)

`$order->status->value`

enum qiymati — model `casts` orqali

`$orders->links()`

sahifalash havolalari

Taqiqlangan

Blade ichida `Order::where(...)` yoki hisob-kitob

TUSHUNISHNI TEKSHIRISH

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

2-bo‘lim yakuni: MVC va Laravel da so‘rov hayot sikli

1

Laravel 12 da marshrut va middleware qaysi faylda sozlanadi?

Savol 1

2

Controller metodiga `OrderService $orders` parametri qanday beriladi?

Savol 2

3

Nima uchun biznes-qoida middleware da yozilmaydi?

Savol 3

03

Repository va Service Layer

Fowler ta’riflari · interfeys va realizatsiya · konteynerda bog‘lash · qachon kerak

Service Layer | Repository | DI | Konteyner

FOWLER, POEAA

Ikki namuna — ikki vazifa

Service Layer — “nima qilinadi”, Repository — “qayerdan olinadi”.

Service Layer

Ilova chegarasidagi amallar to‘plami: `OrderService::create()`, `confirm()`, `cancel()`; tranzaksiya va muvofiqlashtirish.

Repository

Domen obyektlariga kolleksiya kabi kirish: `findBySku()`, `lowStock()`; SQL tafsilotini yashiradi.

Active Record

Eloquent: model o‘zini saqlaydi va so‘raydi — oddiy holatlarda repository shart emas.

DI konteyner

Laravel konstruktor parametrlarini turiga qarab avtomatik yaratadi.

REPOSITORY

Interfeys va Eloquent realizatsiyasi

```php
interface ProductRepository
{
    public function findBySku(string $sku): ?Product;
    public function lowStock(int $warehouseId): Collection;
}

final class EloquentProductRepository implements ProductRepository
{
    // findBySku(): return Product::where('sku', $sku)->first();
    public function lowStock(int $warehouseId): Collection
    {
        return Product::query()
            ->join('stocks', 'stocks.product_id', '=', 'products.id')
            ->where('stocks.warehouse_id', $warehouseId)
            ->whereRaw('stocks.quantity - stocks.reserved < min_stock')
            ->get(['products.*']);
    }
}
```

▶ NATIJA

```
select products.* from products
inner join stocks on stocks.product_id = products.id
where stocks.warehouse_id = ?
and stocks.quantity - stocks.reserved < min_stock
```

interface ProductRepository

servislar shu abstraksiyaga bog‘lanadi

?Product

topilmasa `null` — qaror servisda qabul qilinadi

Collection

natija — tayyor kolleksiya, `Builder` emas

whereRaw

ustunlarni solishtirish; foydalanuvchi qiymati bu yerga qo‘shilmaydi

SERVICE LAYER

Konstruktor in’ektsiyasi va konteynerda bog‘lash

```php
// app/Providers/AppServiceProvider.php
class AppServiceProvider extends ServiceProvider
{
    public function register(): void
    {
        $this->app->bind(
            ProductRepository::class,
            EloquentProductRepository::class,
        );
    }
}
final class OrderService  // app/Services/OrderService.php
{
    public function __construct(
        private ProductRepository $products,
        private StockService $stock,
    ) {}
}
```

▶ NATIJA

```
app(OrderService::class)
→ new OrderService(
    new EloquentProductRepository(),
    new StockService(...)
  )
```

$this->app->bind

interfeys → realizatsiya: har safar yangi obyekt

singleton

`$this->app->singleton()` — bitta umumiy obyekt

private ProductRepository

konstruktor xossasi (PHP 8): e’lon va qiymat bir qatorda

StockService

aniq sinf — konteyner uni bog‘lashsiz ham yarata oladi

TEST DUBLI

Servis bazasiz ishlaydi

```php
interface ProductRepository
{
    public function price(string $sku): ?int;
}
final class FakeProducts implements ProductRepository
{
    private array $rows = ['UN-001' => 15000, 'SH-002' => 9000];
    public function price(string $sku): ?int
    {
        return $this->rows[$sku] ?? null;
    }
}
function narx(ProductRepository $repo, string $sku, int $qty): int
{
    $p = $repo->price($sku) ?? throw new DomainException("$sku yo‘q");
    return $p * $qty;
}
echo narx(new FakeProducts(), 'SH-002', 3), "\n";
```

▶ NATIJA

```
27000
```

FakeProducts

interfeysni amalga oshiruvchi soxta sinf — xotirada

`ProductRepository $repo`

funksiya qaysi realizatsiya kelishini bilmaydi

?? throw

PHP 8: `throw` ifoda sifatida

QACHON KERAK

Repository: foydali va ortiqcha

✓ Foydali

- Murakkab, qayta ishlatiladigan so‘rovlar (`lowStock`)
- Manba almashishi mumkin (baza → 1C API)
- Servisni bazasiz unit-test qilish
- Bir nechta modeldan yig‘iladigan o‘qish modeli

✗ Ortiqcha

- Har bir model uchun `find`, `all`, `save` nusxasi
- `Builder` qaytaradigan “yupqa qobiq”
- Oddiy ma’lumotnoma CRUD i
- Biznes-qoidalar repository ichida

Qoida: avval Eloquent ni to‘g‘ridan-to‘g‘ri servisda ishlating; so‘rov murakkablashsa — repository ga chiqaring.

TUSHUNISHNI TEKSHIRISH

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

3-bo‘lim yakuni: Repository va Service Layer

1

Service Layer va Repository ning vazifasi qanday farq qiladi?

Savol 1

2

Interfeysni realizatsiyaga qayerda va qanday bog‘laymiz?

Savol 2

3

Nima uchun repository `Builder` emas, tayyor natija qaytarishi kerak?

Savol 3

MA’RUZADAGI MASHQ

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

2-topshiriq: Xotiradagi ProductRepository

Testlar uchun mahsulotlarni xotirada saqlaydigan repository yozing. `save($p)` mahsulotni (`sku`, `name`, `price`, `min_stock`, `stock`) saqlaydi; bir xil `sku` qayta saqlansa — yangilanadi. `sku` formati `AA-000` (2 katta harf, chiziqcha, 3 raqam) bo‘lmasa yoki `price < 0` bo‘lsa — `InvalidArgumentException`. `findBySku()` — mahsulot yoki `null`; `lowStock()` — `stock < min_stock` bo‘lgan mahsulotlar SKU lari (alifbo tartibida); `count()` — mahsulotlar soni.

Imzo: `final class InMemoryProductRepository`

Qaytaradi: `findBySku(): ?array`, `lowStock(): list<string>`, `count(): int`.

MISOL · CHAQIRUV

```
(function () { $r = new InMemoryProductRepository(); $r->save(['sku' => 'UN-001', 'name' => 'Un', 'price' => 15000, 'min_stock' => 10, 'stock' => 4]); return $r->findBySku('UN-001')['name']; })()
```

MISOL · NATIJA

```
"Un"
```

Telefon kamerasi bilan skanerlang

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

04

DTO va Unit of Work

Ma’lumot uzatish obyekti · readonly xossalar · tranzaksiya chegarasi · DB::transaction

DTO | readonly | Unit of Work | Tranzaksiya

IKKI NAMUNA

DTO va Unit of Work

DTO ma’lumotni qatlamlar orasida tashiydi, Unit of Work esa o‘zgarishlarni bitta tranzaksiyaga yig‘adi.

DTO

Turlari aniq, o‘zgarmas obyekt: `OrderData`, `OrderItemData` — `$request->all()` o‘rniga.

readonly

PHP 8.1: xossa faqat konstruktorda beriladi, keyin o‘zgartirish — `Error`.

Unit of Work

O‘zgargan obyektlarni yig‘ib, bitta tranzaksiyada yozadi (Doctrine `flush()`).

DB::transaction

Laravel da ish birligi chegarasi: hammasi yoki hech narsa.

APP/DTO

OrderData — so‘rovdan DTO

```php
final class OrderData
{
    public function __construct(
        public readonly int $counterpartyId,
        public readonly int $warehouseId,
        public readonly array $items,  // list<OrderItemData>
        public readonly string $currency = 'UZS',
    ) {}
    public static function fromRequest(StoreOrderRequest $r): self
    { return self::fromArray($r->validated()); }

    public static function fromArray(array $v): self
    {
        return new self((int) $v['counterparty_id'],
            (int) $v['warehouse_id'],
            array_map(OrderItemData::fromArray(...), $v['items']));
    }
}
```

▶ NATIJA

```
OrderData {
  counterpartyId: 7, warehouseId: 1, currency: "UZS",
  items: [OrderItemData { sku: "UN-001", quantity: 5, price: 15000
}]
}
```

`validated()`

faqat validatsiyadan o‘tgan maydonlar — `all()` emas

`(int)`

formadan satr keladi: `"7"` → `7`

`fromArray(...)`

PHP 8.1 birinchi darajali callable sintaksisi

fromRequest → fromArray

forma, API va import bitta kirish nuqtasidan foydalanadi

readonly DTO: yaratish va o‘zgartirishga urinish

```php
final class OrderItemData
{
    public function __construct(
        public readonly string $sku,
        public readonly int $quantity,
    ) {}
    public static function fromArray(array $a): self
    {
        return new self($a['sku'], (int) $a['quantity']);
    }
}
$i = OrderItemData::fromArray(['sku' => 'UN-001', 'quantity' => '5']);
echo $i->sku, ' x ', $i->quantity, "\n";
try {
    $i->quantity = 50;
} catch (Error $e) {
    echo $e->getMessage(), "\n";
}
```

▶ NATIJA

```
UN-001 x 5
Cannot modify readonly property OrderItemData::$quantity
```

`(int) $a['quantity']`

satr `'5'` butun songa aylandi

public readonly

faqat konstruktor ichida bir marta beriladi

Error

o‘zgartirishga urinish — `Exception` emas, `Error`

TRANZAKSIYA

OrderService::create — bitta ish birligi

```php
public function create(OrderData $data): Order
{
    return DB::transaction(function () use ($data) {
        $order = Order::create([
            'counterparty_id' => $data->counterpartyId,
            'warehouse_id' => $data->warehouseId,
            'status' => OrderStatus::Draft,
            'currency' => $data->currency,
            'created_by' => auth()->id(),
        ]);
        foreach ($data->items as $item) {
            $this->addItem($order, $item);
        }
        $order->update(['total' => $order->items()->sum('total')]);

        return $order;
    }, 3);
}
```

▶ NATIJA

```
BEGIN
insert into orders (...) values (...) returning id
insert into order_items (...) values (...)   ×2
update orders set total = 1120000.00 where id = 45
COMMIT        -- istisno bo‘lsa: ROLLBACK
```

`return DB::transaction`

closure natijasi metod natijasi bo‘ladi — `return` ni unutmang

`OrderStatus::Draft`

yangi buyurtma doim `draft` holatida

`addItem()`

mahsulot topilmasa istisno → butun buyurtma bekor

`, 3)`

deadlock bo‘lsa 3 martagacha qayta urinadi

Hammasi yoki hech narsa

```php
function tranzaksiya(array &$db, callable $fn): void
{
    $nusxa = $db;
    try {
        $fn($nusxa);
        $db = $nusxa;
        echo "COMMIT\n";
    } catch (DomainException $e) {
        echo "ROLLBACK: {$e->getMessage()}\n";
    }
}
$db = ['orders' => 0, 'reserved' => 0];
tranzaksiya($db, function (array &$d) { $d['orders']++; });
tranzaksiya($db, function (array &$d) {
    $d['reserved'] += 5;
    throw new DomainException('Qoldiq yetarli emas');
});
echo json_encode($db);
```

▶ NATIJA

```
COMMIT
ROLLBACK: Qoldiq yetarli emas
{"orders":1,"reserved":0}
```

`$nusxa = $db`

o‘zgarishlar nusxa ustida — bu “ochiq tranzaksiya”

`$db = $nusxa`

faqat muvaffaqiyatda asl holatga yoziladi (COMMIT)

`reserved: 0`

ikkinchi urinishdagi `+= 5` saqlanmadi — ROLLBACK

TUSHUNISHNI TEKSHIRISH

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

4-bo‘lim yakuni: DTO va Unit of Work

1

DTO ning `$request->all()` massividan uchta afzalligini ayting.

Savol 1

2

Eloquent da Unit of Work o‘rniga nima ishlatiladi?

Savol 2

3

Tranzaksiya ichida e-mail yuborish nima uchun xavfli?

Savol 3

MA’RUZADAGI MASHQ

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

3-topshiriq: OrderData DTO

Massivdan o‘zgarmas `OrderData` yarating. Majburiy kalitlar: `counterparty_id`, `warehouse_id`, `items` (bo‘sh bo‘lmagan ro‘yxat); har bir qatorda `sku`, `quantity`, `price`. Raqamlar satr bo‘lib kelishi mumkin — butun songa keltiring. `quantity ≤ 0`, `price < 0`, bir xil `sku` ikki marta yoki kalit yetishmasa — `InvalidArgumentException`. `currency` berilmasa — `'UZS'`. Xossalar `readonly` bo‘lsin: `counterpartyId`, `warehouseId`, `items`, `currency`.

Imzo: `final class OrderData`

Qaytaradi: `total(): int` — Σ `quantity × price`; `toArray(): array` — `['counterparty_id', 'warehouse_id', 'currency', 'items', 'total']` tartibida, `items` qatorlari `['sku', 'quantity', 'price']`.

MISOL · CHAQIRUV

```
OrderData::fromArray(['counterparty_id' => '7', 'warehouse_id'
=> 1, 'items' => [['sku' => 'UN-001', 'quantity' => '2', 'price'
=> 15000]]])->toArray()
```

MISOL · NATIJA

```
{"counterparty_id":7,"warehouse_id":1,"currency":"UZS","items":[
{"sku":"UN-001","quantity":2,"price":15000}],"total":30000}
```

Telefon kamerasi bilan skanerlang

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

05

Loyiha strukturasi

Laravel 12 skeleti · bootstrap/app.php · tur va modul bo‘yicha guruhlash · Artisan generatorlari

Struktura | Laravel 12 | Artisan | PSR-4

DARAXT

OmborPro papka strukturasi

```
omborpro/
├── app/
│   ├── DTO/              OrderData, OrderItemData
│   ├── Enums/            OrderStatus
│   ├── Exceptions/       InsufficientStockException
│   ├── Http/
│   │   ├── Controllers/  OrderController
│   │   └── Requests/     StoreOrderRequest
│   ├── Models/           Product, Order, Stock ...
│   ├── Providers/        AppServiceProvider
│   ├── Repositories/     ProductRepository
│   └── Services/         OrderService, StockService
├── bootstrap/app.php     marshrut, middleware, istisnolar
├── database/             migrations, factories, seeders
├── public/index.php      yagona kirish nuqtasi
├── routes/               web.php, console.php
└── tests/                Feature, Unit
```

▶ NATIJA

PSR-4: App\Services\OrderService → app/Services/OrderService.php

app/Services

biznes-jarayonlar — har bir amal bitta metod

app/DTO

qatlamlar orasidagi ma’lumot obyektlari

public/

veb-serverning ildizi — boshqa papkalar tashqaridan ko‘rinmaydi

vendor/

Composer paketlari — `.gitignore` da

LARAVEL 11

bootstrap/app.php — markaziy sozlash

```php
use Illuminate\Foundation\Application;
use Illuminate\Foundation\Configuration\Exceptions;
use Illuminate\Foundation\Configuration\Middleware;

return Application::configure(basePath: dirname(__DIR__))
    ->withRouting(
        web: __DIR__ . '/../routes/web.php',
        commands: __DIR__ . '/../routes/console.php',
        health: '/up',
    )
    ->withMiddleware(function (Middleware $middleware) {
        //
    })
    ->withExceptions(function (Exceptions $exceptions) {
        //
    })->create();
```

▶ NATIJA

```
GET /up → 200 OK  (health check marshruti)
php artisan install:api → api: __DIR__ . '/../routes/api.php'
qo‘shiladi
```

withRouting

marshrut fayllari va health check manzili

health: '/up'

monitoring va Docker uchun “tirikmisan?” sahifasi

withMiddleware

global va guruh middleware lari shu yerda

withExceptions

istisnolarni loglash va javobga aylantirish

GURUHLASH

Tur bo‘yicha va modul bo‘yicha

Tur bo‘yicha (OmborPro)

- `app/Services`, `app/Models`, `app/DTO`
- Laravel hujjatlari va generatorlariga mos
- Kichik va o‘rta loyiha uchun sodda
- Kamchilik: modul chegarasi ko‘rinmaydi

Modul (soha) bo‘yicha

- `app/Domain/Sales/{Models,Services,DTO}`
- Modul chegaralari aniq — modulli monolit
- Katta jamoa va ko‘p modulli tizim uchun
- Kamchilik: generatorlar uchun yo‘lni qo‘lda berish

Qaysi birini tanlasangiz ham — `README.md` da yozib qo‘ying va izchil bo‘ling.

ARTISAN

Generatorlar bilan fayl yaratish

```
php artisan make:model Product -mfs
php artisan make:controller OrderController --resource
php artisan make:request StoreOrderRequest
php artisan make:enum Enums/OrderStatus --string
php artisan make:interface Repositories/ProductRepository
php artisan make:class Services/OrderService
php artisan make:class DTO/OrderData
```

▶ NATIJA

```
INFO  Model [app/Models/Product.php] created successfully.
INFO  Migration
[database/migrations/2026_09_14_..._create_products_table.php]
created successfully.
INFO  Enum [app/Enums/OrderStatus.php] created successfully.
INFO  Class [app/Services/OrderService.php] created successfully.
```

`-mfs`

migratsiya, fabrika va seeder birga yaratiladi

`--resource`

index, create, store, show, edit, update, destroy metodlari

`--string`

satr qiymatli (backed) enum

`make:class`

Laravel 12: oddiy sinf — servis va DTO uchun

TUSHUNISHNI TEKSHIRISH

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

5-bo‘lim yakuni: Loyiha strukturasi

1

Laravel 12 da `routes/api.php` qanday paydo bo‘ladi?

Savol 1

2

PSR-4 avtoyuklash `App\DTO\OrderData` sinfini qaysi fayldan qidiradi?

Savol 2

3

Tur bo‘yicha va modul bo‘yicha guruhlashdan qaysi biri kichik jamoa uchun sodda va nima uchun?

Savol 3

XULOSA

Asosiy xulosalar

1 Qatlamlar

Taqdimot → servis → domen → ma’lumot; bog‘liqlik faqat yuqoridan pastga.

2 MVC

Taqdimot namunasi: Model, View, Controller; biznes-jarayon esa servisda.

3 Service Layer

Har bir biznes-amal — servisning bitta metodi va bitta tranzaksiya.

4 Repository

Murakkab o‘qish va manbani almashtirish uchun; har bir model uchun shart emas.

5 DTO

readonly xossali, turlari aniq obyekt — `$request->all()` o‘rniga.

6 Unit of Work

Laravel da `DB::transaction()`: hammasi yoki hech narsa; tashqi ta’sirlar COMMIT dan keyin.

TAKRORLASH

Yakunlashdan oldin sakkizta savol

1. Vazifalarni ajratish tamoyili nimani talab qiladi?
2. Laravel 12 da middleware qayerda sozlanadi?
3. Route model binding nima?
4. Eloquent qaysi namunani amalga oshiradi?
5. Interfeysni realizatsiyaga qanday bog‘laymiz?
6. readonly xossani o‘zgartirsa nima bo‘ladi?
7. `DB::transaction($fn, 3)` dagi 3 nima?
8. Servis nima qaytarishi kerak?

AMALIY MASHG‘ULOT

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

4 **.env faylini o‘qish**
Oson · 10 ball

5 **PDO DSN satrini yasash**
O‘rta · 20 ball

6 **Docker Compose: band portlar**
O‘rta · 20 ball

7 **Xizmatlarni ishga tushirish tartibi**
Qiyin · 30 ball

8 **Composer versiya cheklovi**
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

Buyurtmani tasdiqlash amali uchun controller, servis va model o‘rtasida vazifalarni qanday taqsimlaysiz?

2

OmborPro da qaysi joyda repository foydali, qaysi joyda ortiqcha deb hisoblaysiz?

3

Bugungi namunalardan qaysi biri siz uchun eng noaniq qoldi?