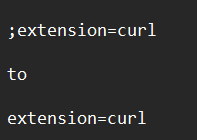


How to enabled : 

This is the extension that needs to enabled in your php.ini.

Laravel snippets, php extension pack, Php intelephense, php namespace resolver, Laravel extension pack

These are need to be downloaded in the extension of visual studio code

Installation of Laravel needs composer, php , and git to be working, if not it going to get an error. If all that setting are ready you can start to create your project kasir.

*\*you can change the name of all that going to i use in the documentation so be carefull not to get eror.*

**START**

composer create-project laravel/laravel fillament-pertama

cd fillament-pertama

composer require filament/filament:"^3.2" -W

php artisan filament:install --panels

* The ID is admin

php artisan vendor:publish --tag=filament-config

composer require flowframe/laravel-trend

* This is for downloading the chart content

# User ( pengguna )

in database\migrations\ 2014\_10\_12\_000000\_create\_users\_table.php

* Tambahkan setelah password

$table->enum('level', [0, 1])->default(0);

ubah env ke database yang baru

DB\_DATABASE=fillament-pertama

php artisan migrate

* yes for creating new database

php artisan make:filament-user

* Name = admin
* Email = [admin@gmail.com](mailto:admin@gmail.com)
* Password = admin

php artisan filament:resource User --simple

in app\Filament\Resources\UserResource.php

* Add the protected fillable for level in app\Models\User.php

    protected $fillable = [

        'name',

        'email',

        'level',

        'password',

    ];

* in app\Filament\Resources\UserResource.php add this

use App\Filament\Resources\UserResource\Pages;

use App\Models\User;

use Filament\Forms\Form;

use Filament\Resources\Resource;

use Filament\Tables;

use Filament\Tables\Table;

use Filament\Tables\Columns\TextColumn;

use Filament\Forms\Components\TextInput;

use Filament\Forms\Components\Select;

* in app\Filament\Resources\UserResource.php the scema for the form

    public static function form(Form $form): Form

    {

        return $form

            ->schema([

                TextInput::make('name')

                    ->required()

                    ->maxLength(255),

                TextInput::make('email')

                    ->label('Email Address')

                    ->required()

                    ->maxLength(255),

                TextInput::make('password')

                    ->required()

                    ->password()

                    ->revealable()

                    ->visibleOn('create')

                    ->maxLength(255),

                Select::make('level')

                    ->options([

                        0 => 'Admin',

                        1 => 'Pegawai',

                    ])

                    ->required(),

            ]);

    }

* in app\Filament\Resources\UserResource.php the column in table function

            ->columns([

                TextColumn::make('name')->label('Name')->searchable(),

                TextColumn::make('email')->label('Email Address')

                ->searchable(),

                TextColumn::make('level')->label('level')->searchable()

                ->getStateUsing(function ($record) {

                    return $record->level == 0 ? 'Admin' : 'Pegawai';

                }),

            ])

* Add this function in app\Filament\Resources\UserResource.php to prevent access from pegawai and only can be accessed by admin

    public static function canAccess(): bool

    {

        return auth()->user()->level == 0;

    }

* Add this for managing the tab in panel

    protected static ?string $navigationGroup = 'Users';

    public static ?string $navigationLabel = 'pengguna';

# Product ( Produk )

php artisan make:model Product -m

* Tambahkan field ini di di database\migrations\0000\_00\_00\_000000\_create\_products\_table.php

            $table->id('ProdukID');

            $table->string('NamaProduk');

            $table->decimal('Harga', 10, 2);

            $table->integer('Stok');

            $table->timestamps();

* php artisan migrate
* in the app\Models\Product.php add this

    protected $fillable = [

        'NamaProduk',

        'Harga',

        'Stok',

    ];

    protected $primaryKey = 'ProdukID';

* php artisan make:filament-resource Product --simple
* In app\Filament\Resources\ProductResource.php in the function form scema add

                TextInput::make('ProdukID')

                ->label('Product ID')

                ->hiddenOn('create'),

                TextInput::make('NamaProduk')

                    ->required()

                    ->label('Nama Produk'),

                TextInput::make('Harga')

                    ->numeric()

                    ->required()

                    ->label('Harga')

                    ->step(1000),

                TextInput::make('Stok')

                    ->numeric()

                    ->required()

                    ->label('Stock')

                    ->step(1),

* And in the column in table function add

            ->columns([

                TextColumn::make('ProdukID')

                    ->label('Product ID'),

                TextColumn::make('NamaProduk')

                    ->label('Nama Produk')

                    ->searchable(),

                TextColumn::make('Harga')

                    ->numeric()

                    ->label('Harga'),

                TextColumn::make('Stok')

                    ->numeric()

                    ->label('Stock'),

            ])

* in the function table add DeleteAction to actions

            ->actions([

                Tables\Actions\EditAction::make(),

                Tables\Actions\DeleteAction::make(),

            ])

* don’t forget to managing this page on tab in the panel

    protected static ?string $navigationGroup = 'Market';

    public static ?string $navigationLabel = 'Produk';

# Customer ( pelanggan )

php artisan make:model customer -m

* Tambahkan field ini di di database\migrations\0000\_00\_00\_000000\_create\_customers\_table.php

            $table->id('PelangganID');

            $table->string('NamaPelanggan');

            $table->string('Alamat');

            $table->string('NomorTelepon');

            $table->timestamps();

* php artisan migrate
* php artisan make:filament-resource Customer --simple
* in the app\Models\customer.php add this

    protected $fillable = [

        'NamaPelanggan',

        'Alamat',

        'NomorTelepon',

    ];

    protected $primaryKey = 'PelangganID';

* in app\Filament\Resources\CustomerResource.php in the function form schema add

            ->schema([

                TextInput::make('NamaPelanggan')

                    ->label('Customer Name')

                    ->required(),

                TextInput::make('Alamat')

                    ->label('Address')

                    ->required(),

                TextInput::make('NomorTelepon')

                    ->label('Phone Number')

                    ->required(),

            ]);

* and in the function table column add

            ->columns([

                TextColumn::make('PelangganID')

                ->searchable()

                ->label('Customer ID'),

                TextColumn::make('NamaPelanggan')

                    ->searchable()

                    ->label('Customer Name'),

                TextColumn::make('Alamat')

                    ->searchable()

                    ->label('Address'),

                TextColumn::make('NomorTelepon')

                    ->searchable()

                    ->label('Phone Number'),

                ])

* don’t forget to managing this page on tab in the panel

    protected static ?string $navigationGroup = 'Market';

    public static ?string $navigationLabel = 'Pembeli';

* in the function table add DeleteAction to actions

            ->actions([

                Tables\Actions\EditAction::make(),

                Tables\Actions\DeleteAction::make(),

            ])

# Sale ( penjualan )

php artisan make:model Sale -m

* heres is the fields in database\migrations\0000\_00\_00\_000000\_create\_sales\_table.php

            $table->id('PenjualanID');

            $table->dateTime('TanggalPenjualan');

            $table->decimal('TotalHarga', 10, 2);

            $table->unsignedBigInteger('PelangganID');

            $table->foreign('PelangganID')->references('PelangganID')->on('customers')->onDelete('cascade');

            $table->timestamps();

php artisan make:model SaleDetail -m

* heres the fields in database\migrations\0000\_00\_00\_000000\_create\_sale\_details\_table.php

            $table->id('DetailID');

            $table->unsignedBigInteger('PenjualanID');

            $table->foreign('PenjualanID')->references('PenjualanID')->on('sales')->onDelete('cascade');

            $table->unsignedBigInteger('ProdukID');

            $table->foreign('ProdukID')->references('ProdukID')->on('products')->onDelete('cascade');

            $table->integer('JumlahProduk');

            $table->decimal('Subtotal', 10, 2);

            $table->timestamps();

php artisan migrate

* in app\Models\SaleDetail.php add this and also add a function

    protected $fillable = [

        'PenjualanID',

        'ProdukID',

        'JumlahProduk',

        'Subtotal',

    ];

    protected $primaryKey = 'DetailID';

    public function penjualan(){

        return $this->hasMany(Sale::class, 'PenjualanID', 'PenjualanID');

    }

    public function produk(){

        return $this->belongsTo(Product ::class, 'ProdukID', 'ProdukID');

    }

* in app\Models\Sale.php add this and the function

    protected $fillable = [

        'TotalHarga',

        'PelangganID',

        'TanggalPenjualan',

    ];

    protected $primaryKey = 'PenjualanID';

    protected static function boot()

    {

        parent::boot();

        static::creating(function ($penjualan) {

            $penjualan->tanggalpenjualan = now()->toDateString();

        });

    }

    protected $guarded = [];

    public function pelanggan()

    {

        return $this->belongsTo(customer::class, 'PelangganID', 'PelangganID');

    }

    public function produk()

    {

        return $this->belongsTo(Product::class);

    }

    public function saledetails()

    {

        return $this->hasMany(SaleDetail::class, 'PenjualanID', 'PenjualanID');

    }

php artisan make:filament-resource Sale

* in the app\Filament\Resources\SaleResource.php add this to the table function in the columns

            ->columns([

            TextColumn::make('PenjualanID')

                ->searchable()

                ->label('Penjualan ID'),

            TextColumn::make('TanggalPenjualan')

                ->searchable()

                ->dateTime('d M Y')

                ->label('Tanggal Penjualan'),

            TextColumn::make('pelanggan.NamaPelanggan')

                ->searchable()

                ->label('nama pelanggan'),

            TextColumn::make('pelanggan.Alamat')

                ->searchable()

                ->label('Alamat'),

            TextColumn::make('saledetails.produk.NamaProduk')

                ->searchable()

                ->label('nama product'),

            TextColumn::make('saledetails.JumlahProduk')

                ->searchable()

                ->label('jumlah product'),

            TextColumn::make('TotalHarga')

                ->label('Total Harga'),

            ])

* in the app\Filament\Resources\SaleResource.php in the form function add this

            ->schema([

                Wizard::make([

                    Step::make('Detail Pembelian')

                        ->schema([

                            Select::make('PelangganID')

                                ->preload()

                                ->searchable()

                                ->required()

                                ->relationship('pelanggan', 'Namapelanggan'),

                        ]),

                    Step::make('Order Produk')

                        ->schema([

                            Repeater::make('produk')

                            ->required()

                            ->label('Pilih Produk')

                            ->columns(3)

                            ->defaultItems(1)

                            ->live()

                            ->relationship('saledetails')

                            ->schema([

                                Select::make('ProdukID')

                                    ->preload()

                                    ->required()

                                    ->searchable()

                                    ->relationship('produk', 'NamaProduk')

                                    ->live()

                                    ->afterStateUpdated(fn ($state, Set $set) => $set('Subtotal', Product::find($state)?->Harga ?? 0)),

                                    TextInput::make('JumlahProduk')

                                    ->numeric()

                                    ->default(1)

                                    ->reactive()

                                    ->live()

                                    ->disabled(fn ($state, $get) => is\_null($get('ProdukID')))

                                    ->afterStateUpdated(function ($state, $set, $get) {

                                        $productModel = Product::find($get('ProdukID'));

                                        if ($state > $productModel->Stok) {

                                            $state = $productModel->Stok;

                                            Notification::make()

                                                ->title('Not enough stock')

                                                ->body('The quantity you requested for product ' . $productModel->NamaProduk . ' is not available. The current stock: ' . $state)

                                                ->seconds(10)

                                                ->danger()

                                                ->send();

                                        }

                                        $set('Subtotal', $productModel->Harga \* $state);

                                    })

                                    ->rules('required'),

                                TextInput::make('Subtotal')

                                    ->disabled()

                                    ->numeric()

                                    ->dehydrated()

                                    ->prefix('Rp.')

                                    ->required(),

                                ]),

                                TextInput::make('TotalHarga')

                                ->required()

                                ->default(0)

                                ->prefix('Rp.')

                                ->numeric()

                                ->inputMode('decimal')

                                ->mask(fn (Get $get)=>collect($get('produk'))->pluck('Subtotal')->sum()),

                        ]),

                ])->columnSpanFull()

            ]);

* in app\Filament\Resources\SaleResource\Pages\CreateSale.php add this function to decrementing product that got inputed and the function to got redirected to index after done creating record

    protected function afterCreate(): void

    {

        foreach ($this->record->saledetails as $saledetail) {

            $productModel = Product::find($saledetail['ProdukID']);

            if ($productModel->Stok < $saledetail['JumlahProduk']) {

                // Not enough stock available

                Notification::make()

                    ->title('Sales Not Created')

                    ->body('Not enough stock available for product:' . $productModel->NamaProduk)

                    ->seconds(10)

                    ->danger()

                    ->send();

                    $this->record->delete();

                    return;

            }

            $productModel->Stok -= $saledetail['JumlahProduk'];

            $productModel->save();

        }

    }

    protected function getRedirectUrl():string{

        return $this->getResource()::getUrl('index');

    }

* also add this in app\Filament\Resources\SaleResource.php new function to prevent admin from inserting sale

    public static function canAccess(): bool

    {

        return auth()->user()->level == 1;

    }

* don’t forget to add this to to manage the name in the tab panel

    protected static ?string $navigationGroup = 'Market';

    public static ?string $navigationLabel = 'Penjualan';

* and add this code to the function table in actions

            ->actions([

                Tables\Actions\EditAction::make(),

                Tables\Actions\ViewAction::make(),

            ])

# Chart Widget ( chart di admin panel )

php artisan make:filament-widget SaleChart --chart

skip the resource [enter]

create this on admin

the chart is whatever you want ( I recommend using 3 that is line chart)

* in app\Filament\Widgets\SaleChart.php add this to the getData function

    protected function getData(): array

    {

        $data = Trend::model(SaleDetail::class)

            ->between(

                start: now()->startOfMonth(),

                end: now()->endOfMonth(),

            )

            ->perDay()

            ->count();

        return [

            'datasets' => [

                [

                    'label' => 'Sale Detail',

                    'data' => $data->map(fn (TrendValue $value) => $value->aggregate),

                ],

            ],

            'labels' => $data->map(fn (TrendValue $value) => $value->date),

        ];

    }

* in app\Providers\Filament\AdminPanelProvider.php in widgets remove filamentinfowidget to remove the filament watermark on admin panel

            ->widgets([

                Widgets\AccountWidget::class,

                // Widgets\FilamentInfoWidget::class,

            ])

php artisan serve

* when in 127.0.0.1:8000 go to 127.0.0.1:8000/admin
* insert email : [admin@gmail.com](mailto:admin@gmail.com)
* and password : admin
* first go to users in pengguna tab create new user then the level is pegawai
* then logout and login with the new created user
* then insert pembeli and produk in market and then the penjualan
* And you all set ready to go!