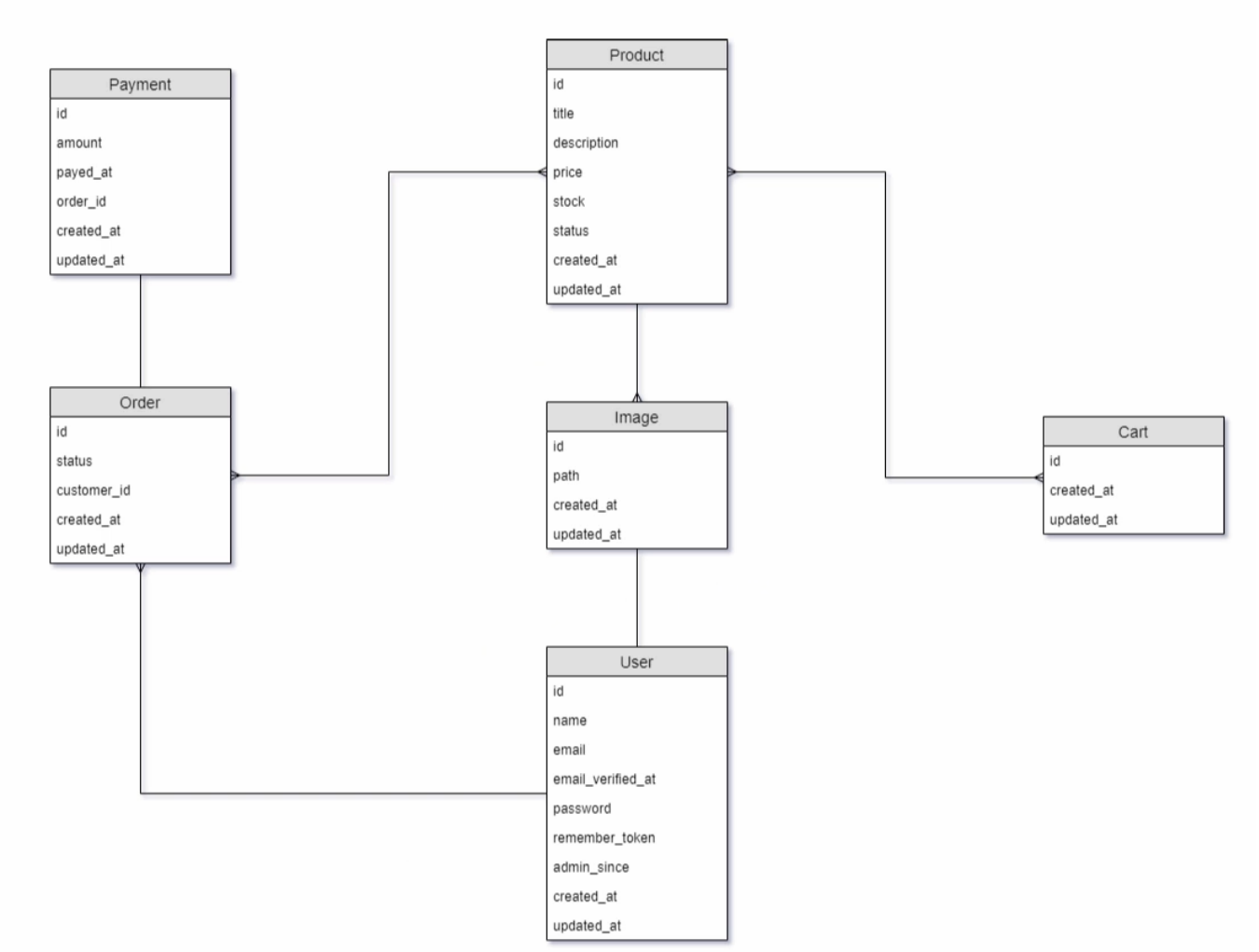
62. Aprende sobre relaciones polimórficas muchos a muchos

Definición

Es evidente que un **Cart** puede tener varios **Product** y varios **Product** pueden estar en un mismo **Cart**. De igual manera ocurre con **Product** ↔ **Order**.



¿Dónde implemento la relación polimórfica?

La ambigüedad, en este caso, está en **Product**. Es él quien tendrá que especificar si se encuentra en un **Cart** o una **Order**.

Relaciones polimórficas m:n en el proyecto

- Cart ↔ Product.
- Order ↔ Product.

Laravel: crear las relaciones

Borrar migraciones

Borramos una de las dos migraciones CreateCartProductTable o CreateOrderProductTable, y trabajamos sobre la otra, renombrándola

```
CreateProductablesTable
```

Importante que sea en **plural**, y que **cambiemos también el nombre de archivo** a

```
[fecha_creacion]_create_productables_table.php
```

En **CreateProductablesTable**:

```php
public function up()
{
    Schema::create('productables', function (Blueprint $table) {
        // Al crear morphs() un productable_id, este atributo
        // deja de ser necesario.
        // $table->bigInteger('order_id')->unsigned();
        $table->bigInteger('product_id')->unsigned();
        $table->bigInteger('quantity')->unsigned();
        $table->morphs('productable');

        // $table->foreign('order_id')->references('id')->on('orders');
        $table->foreign('product_id')->references('id')->on('products');
    });
}
```

En **Product**:

```php
public function carts()
{
    return $this->morphedByMany(Cart::class, 'productable')
            ->withPivot('quantity');
}

public function orders()
{
    return $this->morphedByMany(Order::class, 'productable')
            ->withPivot('quantity');
}
```

En **Cart** y **Order**:

```php
public function products()
{
    return $this->morphToMany(Product::class, 'productable')
            ->withPivot('quantity');
}
```

Tinker: probar las relaciones

Primero debemos ejecutar de nuevo todas las #migraciones para actualizar los cambios en la base de datos:

```
php artisan migrate:fresh --seed
```

```
C:\Users\llemi\Desktop\Lunk\Laravel\ola>php artisan tinker
Psy Shell v0.11.8 (PHP 8.0.9 - cli) by Justin Hileman

// Creamos las instancias necesarias para probar la relación
>>> $product = App\Models\Product::find(3)
=> App\Models\Product {#4558
     id: "3",
     title: "Ab quibusdam velit labore sit.",
     description: "Maiores repudiandae et similique itaque fugiat velit.",
     price: "56.81",
     stock: "7",
     status: "unavailable",
     created_at: "2022-11-12 13:50:33",
     updated_at: "2022-11-12 13:50:33",
   }

>>> $cart = App\Models\Cart::factory()->create()
=> App\Models\Cart {#4590
     updated_at: "2022-11-12 13:51:27",
     created_at: "2022-11-12 13:51:27",
     id: 1,
   }

>>> $user = App\Models\User::factory()->create()
=> App\Models\User {#4593
     name: "Madilyn Corkery",
     email: "wisozk.alfonzo@example.org",
     email_verified_at: "2022-11-12 13:52:02",
     #password: "$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi",
     #remember_token: "fvUl7DPAjb",
     admin_since: "null()",
     updated_at: "2022-11-12 13:52:02",
     created_at: "2022-11-12 13:52:02",
     id: 1,
   }

>>> $order = App\Models\Order::factory()->create(['customer_id' => $user->id])
=> App\Models\Order {#4600
     status: "paid",
     customer_id: 1,
     updated_at: "2022-11-12 13:52:43",
     created_at: "2022-11-12 13:52:43",
     id: 1,
   }

>>> // Agregamos órdenes al producto
>>> $product->orders()->attach([1 => ['quantity' => 5]])
=> null

>>> $product->orders
=> Illuminate\Database\Eloquent\Collection {#4549
     all: [
       App\Models\Order {#4551
         id: "1",
         status: "paid",
         customer_id: "1",
         created_at: "2022-11-12 13:52:43",
         updated_at: "2022-11-12 13:52:43",
         pivot: Illuminate\Database\Eloquent\Relations\MorphPivot {#4564
           product_id: "3",
```

```
            productable_id: "1",
            productable_type: "App\Models\Order",
            quantity: "5",
          },
        },
      ],
    }

// Añadimos el producto que teníamos en la variable $product usando
// su id, al carrito. Pivotamos con la cantidad, que será 7.
>>> $cart->products()->attach([$product->id => ['quantity' => 7]])
=> null

>>> $cart->products
=> Illuminate\Database\Eloquent\Collection {#4603
     all: [
       App\Models\Product {#4605
         id: "3",
         title: "Ab quibusdam velit labore sit.",
         description: "Maiores repudiandae et similique itaque fugiat velit.",
         price: "56.81",
         stock: "7",
         status: "unavailable",
         created_at: "2022-11-12 13:50:33",
         updated_at: "2022-11-12 13:50:33",
         pivot: Illuminate\Database\Eloquent\Relations\MorphPivot {#4587
           productable_id: "1",
           product_id: "3",
           productable_type: "App\Models\Cart",
           quantity: "7",
         },
       },
     ],
   }
```