Model e Registros

Para que o sistema de ordenação funcione corretamente, cada **model** envolvida deve seguir uma estrutura mínima obrigatória.
Abaixo estão descritas as configurações e implementações necessárias para garantir a compatibilidade com a feature.

---

1. **Trait obrigatória**

Toda model que utilizar o sistema de ordenação **deve incluir a trait**:

use App\Traits\SortableTrait;

Essa trait contém os métodos e comportamentos essenciais para manipular a posição dos registros, incluindo funções de ordenação, reordenação após exclusões e cálculo da próxima posição disponível.

---

2. **Evento deleted com reorder()**

É obrigatório que a model implemente o evento **deleted**, chamando o método **reorder()** após a exclusão de um registro.
Isso garante que a ordenação seja recalculada automaticamente, mantendo a sequência correta dos itens.

Exemplo:

```
protected static function boot()
    {
        parent::boot();

        static::deleted(function ($model) {
            $model->reorder(); // executa após deletar com sucesso
        });
    }
```

> **Observação:**
> Se a ordenação possuir níveis (ex: seção → produto), é necessário passar um *filtro* via **Closure** como terceiro parâmetro na função **reorder()**, indicando o escopo onde a reordenação deve ocorrer (por exemplo, dentro da mesma seção).

---

3. **Relacionamentos entre níveis**

Quando o módulo utiliza **ordenação com múltiplos níveis**, cada nível adicional deve possuir uma **relação "belongsTo"** com o nível imediatamente anterior.
Essa relação é obrigatória para que o sistema saiba a qual contexto o registro pertence e consiga filtrar corretamente antes de aplicar a ordenação.

Exemplo:
Na model do **nível 3**, deve existir uma relação com o **nível 2**:

```
public function nivel2() { return $this->belongsTo(NivelDoi::class, 'nivel_2_id'); }
```

---

4. **Colunas obrigatórias na tabela**

A tabela da model deve conter as seguintes colunas:

Coluna	Tipo	Obrigatoriedade	Descrição
order	**integer**	**Obrigatória**	Define a posição do registro dentro do contexto da ordenação.
parent_id	**integer**	Obrigatória **somente** quando o tipo de ordenação é **recursiva** (entre registros da mesma model).	
<nome_da_fk>	**integer**	Obrigatória **quando existe mais de um nível** (não recursiva). Representa a foreign key que liga ao nível anterior.	

> **Exemplo:**
>
> - Em uma ordenação recursiva (ex: categorias com subcategorias), use **parent_id**.
> - Em uma ordenação com múltiplos níveis (ex: seção → produto), use a FK correspondente, como **secao_id**.

> - Em uma ordenação simples (sem níveis), nenhuma FK adicional é necessária.

Revisão #4
Criado 13 outubro 2025 13:36:32
Atualizado 22 maio 2026 21:15:39