<td>{{ $item->products->selling\_price }}</td>  
The line `<td>{{ $item->products->selling\_price }}</td>` in your `checkout.blade.php` view is fetching the `selling\_price` property of a product associated with a cart item (`$item`) using the defined relationship in your `Cart` model. Let's break down how this logic works in detail:

1. \*\*View Rendering\*\*:

- You are rendering a Blade template named `checkout.blade.php`, presumably as part of a checkout page in your Laravel application.

- Inside this Blade template, you have a table cell (`<td>`) where you want to display the selling price of a product.

2. \*\*CheckoutController\*\*:

- In your `CheckoutController`, specifically in the `index` method, you are retrieving cart items associated with the currently authenticated user (`Auth::id()`). You are using the `Cart` model and the `where` method to fetch these cart items.

3. \*\*Relationship Defined in Cart Model\*\*:

- In your `Cart` model, you have defined a relationship method named `products` using Eloquent's `belongsTo` relationship.

- This relationship method establishes a link between a cart item and a product. It indicates that each cart item "belongs to" a product, and the relationship is based on the `prod\_id` column in the `Cart` table, which corresponds to the `id` column in the `Product` table.

4. \*\*Usage in Blade Template\*\*:

- Back in your Blade template (`checkout.blade.php`), you have a variable `$item`, which presumably represents a cart item.

- You are using the `products` relationship to access the associated product for that cart item: `$item->products`.

- Specifically, you are accessing the `selling\_price` property of the associated product using `$item->products->selling\_price`.

5. \*\*How It Works\*\*:

- When you call `$item->products`, Laravel recognizes the relationship method and automatically queries the database to fetch the product associated with the cart item.

- The relationship is established based on the `prod\_id` value in the cart item, which links to the `id` of the product in the `Product` table.

- Once the associated product is fetched, you can access its properties, such as `selling\_price`, as if it were a property of the cart item.

In summary, this code leverages Laravel's Eloquent ORM and the defined relationship between cart items and products to retrieve and display the selling price of a product associated with each cart item on the checkout page. The relationship simplifies the process of fetching related data and makes it easy to work with associated records in your views.