ADD PRODUCT - DATA STRUCTURE EXPLANATION

Nicole Jay | March 29, 2018

Explanation of variants/attribute sets

Product without any variations or attributes
Usage scenario 1 - Gold necklace with pendant. There are no options for this product. One price, one qty value, and one set of specs.

Product with Variants / Attribute Sets
Usage scenario 1 - One variation of a product that must be selected before checking out. Ie, Product is a T- Shirt with Print, you HAVE to pick a size. Small, Medium, Large, or 8,10,12 in order to checkout.
Each option is the same price, and same weight, but the STOCK QTY differs for each size.

Usage scenario 2 - additional options for a product with a default option, ie, Product is Print Artwork, which comes in different sizes. The default size is A4, but you can get it in A3 and A1.
Each size is a different cost, as well as different width, height and weight.

Usage scenario 3 - SET OF ATTRIBUTES, multiple variations of a product that must be selected before checking out. Ie Product is a T-shirt with print, you HAVe to pick a size AND colour.
Example configurations - *3x Size Small in Pink*
5x Size Medium in Pink
0x Size Large in Pink - OUT OF STOCK

10x Size Small in Black
15x Size Meidum in Black
8x Size Large in Black
They all cost the same, and weigh the same, but the STOCK QTY differ for each variation, and you have to choose 2 options, size, and colour, before you can add to cart.

Why do we need specifications at the variant/attributeSet level? Because sometimes the variation of the product will yield different values for dimension and weight as shown above. Sometimes it won't (also shown above), but if we don't accommodate for it then in the scenarios where those specs will differ will inherently restrict the seller from configring their product as such and will force them to create product listings for EACH VARIATION OF A PRODUCT, resulting in more listings they have to maintain individually. We don't want to ask sellers to sign up and pay for our platform but give them a less innovative / cutting edge experience than what they may be getting with other ecommerce platforms. Im short, we don't want to cause configuration hurdles from the onset

When you add to cart, what is actually being added? As far as the database is concerned, the record that is referenced in the checkout, or order history, is the record that contains the price and qty, for the product item, whether thats the default item, or the variant/attribute option(s) that was selected.

If you add a small pink printed t-shit, it is the small pink printed t shirt product item record that is first checked for available qty, and is then added to the cart if it can be added.

If a product has no options, it still has ONE product item record, which holds the default price, qty and specs, and its this record that gets added to the cart as well.

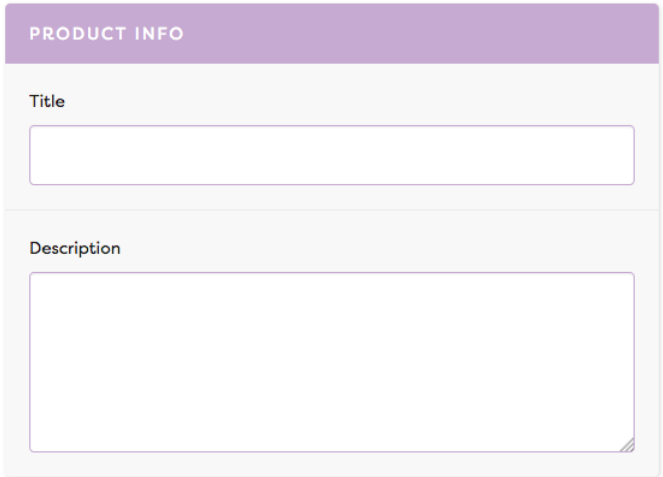
Limitations with the current design? Currently specifications can only be added to the listing itself, not each individual product variation. This would be ok in the case of a product that doesnt have any options, but in the case where there are options, like print size, where dimensions and weight may change, these differing values have no where to be added. We could get around this by just giving a custom attribute option label that displays the dimensions as a string, however we plan to use specifications, particularly dimension and weight, as values when calculating shipping once we bring calculated shipping in, so its important to get the database schema correct now, rather than having to change data models later.

What needs to change? For the most part the add-product page is spot on, we just need to think more about the flow of adding the main listing data, and then redesigning the interface for the default product item, to accommodate for when additional product variations are needed, and when they're not needed so that its easy for the user to navigate. Next pages elobarate on the limitations of the current design, the data models and offers a possible solution.

CURRENT DESIGN - THE PRODUCTID DEPENDENCY

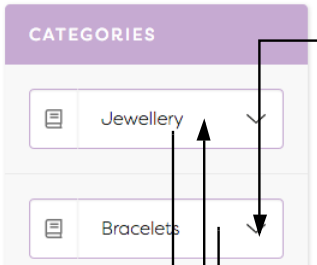
Nicole Jay | March 29, 2018

The 'Product' should be seen as the listing itself. Categories, Tags, & Shipping, all rely on a productID existing and sit in a different database table. The price, sku, qty, specs, attributes and any other variation of the these values also rely on the productID existing, and sit in a different database table (discussed in the next page). The main listing data either needs to be saved as a step 1, where the rest of the data is only accessible after saving the basic listing details, or if we opt for a 'save all at once' design, saved FIRST in the databse before the rest of the data can be saved,. The latter not really being an ideal solution, especially when leveraging the design/layout for EDITING/UPDATING an existing product.



Product (THE ONLY DATA NEEDED TO SAVE TO DB, TO GET A PRODUCTID)
productID (genrated on save)
shopID (pre defined)
title
slug (auto determined)
description
order (auto determined on initial save)
metaTItle* (phase 2)
metaKeyword* (phase 2)
metaDescription* (phase 2)
status (defaults to 'Unpublished' until button is pressed)

This info is on a listing level. Nothing specific to variations, or specifications. Its not what is added to the cart, but holds information that is true for all variatons of the product.



Categories
categoryID
Label
parentID

A shop can have many categories, their initial categories are pre defined by their events system profile categories

ShopCategories
shopID
categoryID

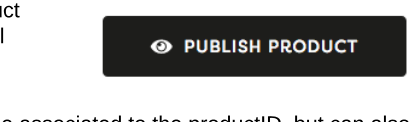
ProductCategories
productID
categoryID

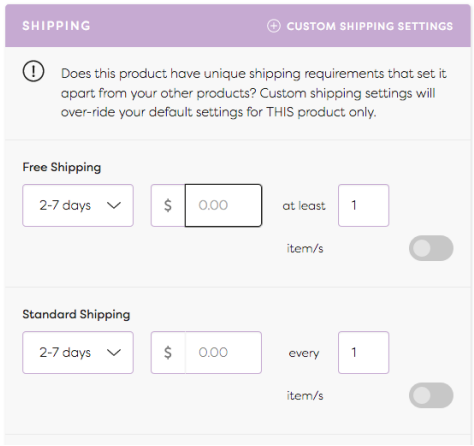
Categories - The categories DISPLAYED in the first option, (Main Category) come from the shop categories table. There may only be one, there may be many.

Subcategories - These cascade down based on the MAIN category selection, where the parentID equals the previous category ID.

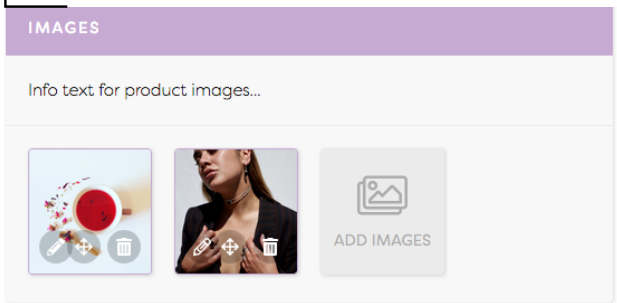
When selected - A reference to the product ID (which is the product listing) is saved agaist each category/sub category selected. In the above example, two records would be saved in ProductCategories: one for jewellery, and one for bracelets. This means this product listing will show up in both of those categories.

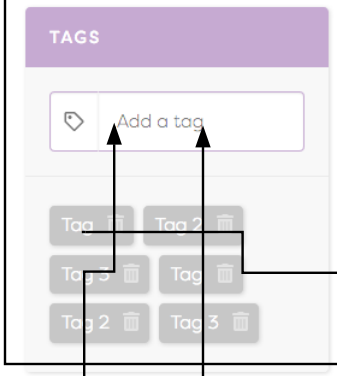
Shipping deserves its own conversation, and maybe its own page to describe the data model, but in this scope, shipping on a product listing level is only reliant on the productID, it is not set for individaul product variations. If shipping on a product level isnt set, then the shop level shipping options will take affect.





All images will be associated to the productID, but can also be referenced in a product item variation, particularly in the case of the default product item record, this image will be the main product image, all other images will just be viewable in the gallery, in the order in which they're specified.





Tags are used on listings, not individual product variations. Tags are general, not specific to a variation/attributeSet. Attributes are filterable and searchable, so adding tags on that level is overkill. Therefor, Tags are saved against the productID and product model. This is a polymorphic relationship in the database, so that we can open up other models to be taggable in future.

Tags
taggable_id (productID)
taggable_type (Product)
tag

A user would start typing in the field and it would autocomplete search for existing tags entered by any user. When selected, a new record is added to the tags table against this productID. If no tag already exists, it is created against this productID.

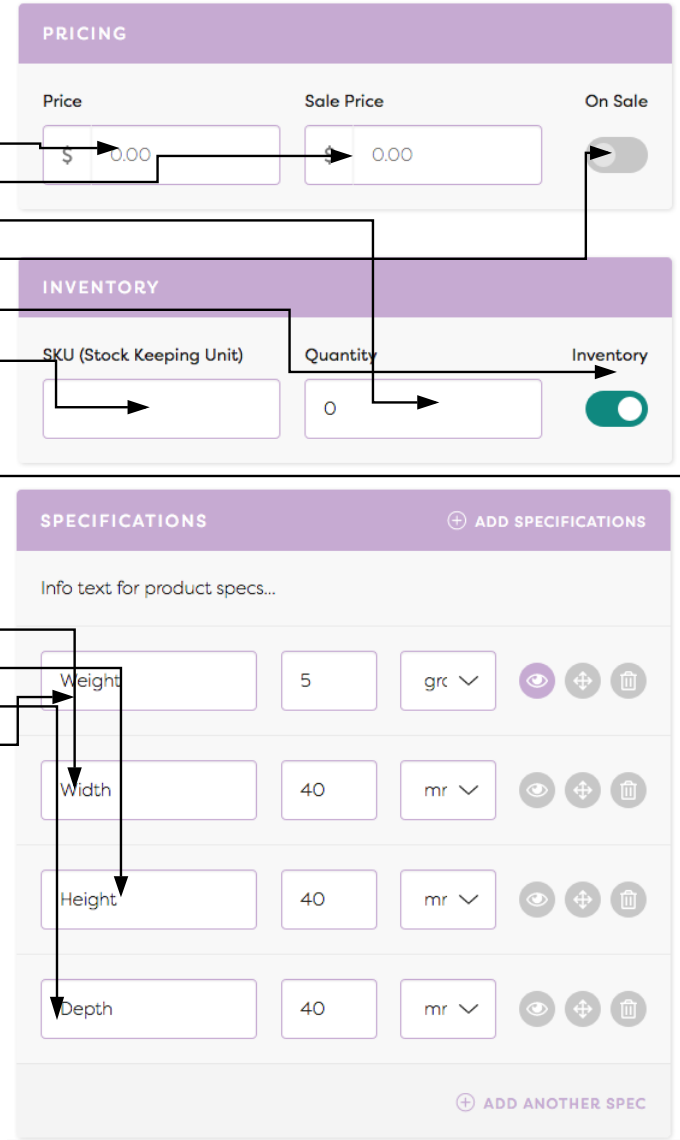
CURRENT DESIGN - THE PRODUCT ITEM(S) & ASSOCIATED DATA

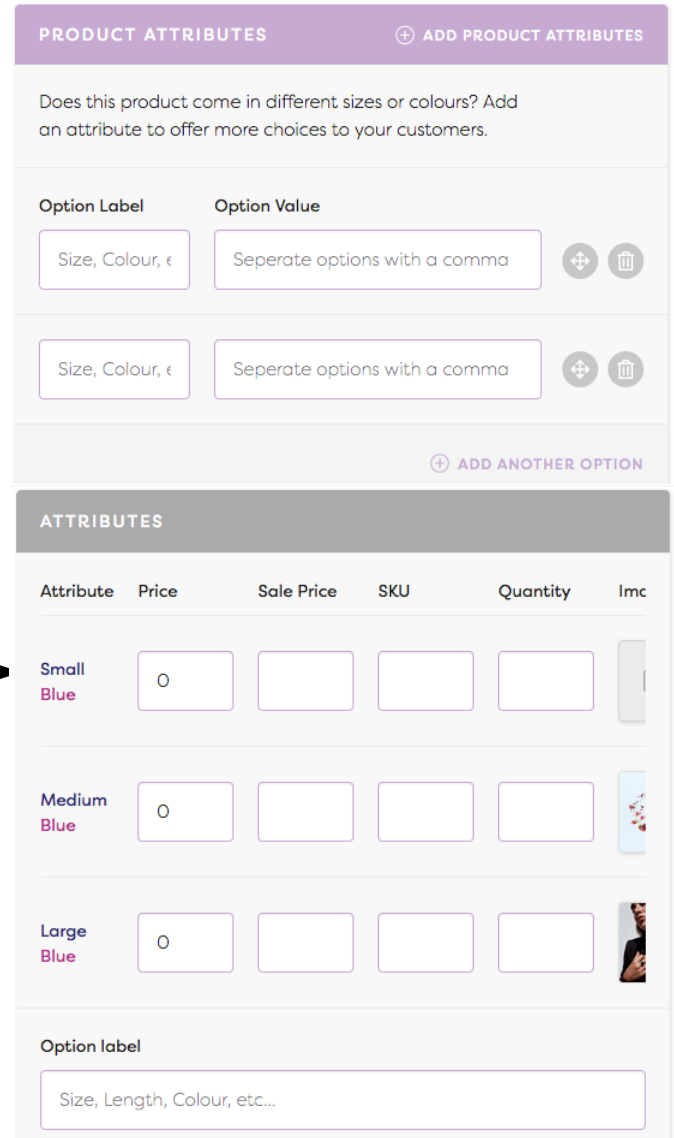
Nicole Jay | March 29, 2018

Look at product item(s) as the actual physical product that will eventually be shipped to someone. This is the entity being sold. In some cases, a product listing may only have one product item that is being sold, or it may have variations, or additional options available that would produce a new physical item. A product item cannot exist without a productID/product listing, and can only be created if a productID is known. A product item also has associated data that defines it's physical properties. This data is associated with the ProductItemID.

In order for a product listing to be published, it has to have at least ONE product item, ie. at least one physical item being sold.

Product Item and associated data Example
Item Info
Original Price - $50
Sale Price - $25
QTY - 100
On sale - No
Inventory Tracking - Yes
SKU
Active on listing - Yes
Default item - Yes
Variant/Attributes
Size - Small
Colour - Blue
Specifications
Width - 40mm
Height - 40mm
Depth - 40mm
Weight - 5 grams
Image
Image from gallery





This design layout would be ok in the scenario where there are no product variations, ie if Size, Colour, etc aren't needed.

But when the seller does choose to configure variations, they only have access to specify the price/inventory/image data and the attributes themselves, and won't be able to configure the specifications for any addtitional variation.

KNOWN REQUIREMENTS
A base product listing must be created first. Logically, at minimum, only the **title** is required to obtain the productID.

A productID, or product listing, must have at least ONE productItemID, or product item (physical product that can be added to a cart) associated to it.

Only when a product has variations will there be more than one productItem associated to a product listing.

Price details, Inventory details, Specifications, and attribute values should be defineable on the product item level.

CURRENT DESIGN RESTRICTIONS
Price, Inventory and Specifications appear to be defineable at the listing level. While ok for products that dont have variations, in products that have variations, this could be confusing.

In the attribute set configuration, price, inventory and image are configurable, but specifications isn't.

POSSIBLE SOLUTION

When you land on the "add product" page, you're met with a simple form. One that asks you to specify your product title, and description if we want to save that in this step too. These should still be editable on the next step, but allows us to get the productID straight up.

On save, this creates the base product record, returns the productID, then the rest of the options become available. Whether thats coming from invisible to visible, or disabled to enabled, thats a design choice, and UX decision.

First, you'll see a box for "product item info", that shows all the price, inventory and spec options. Within this box there could be a prompt "does this product have selectable attributes, like size/colour"?

If no, no further details are required and this item is saved as the one and only productItem. If yes, the option to select your attribute(s) becomes visible. Also, a button for "add another product variation" is visible, which would duplicate the entire first product item box, including price, inventory and specs, pre populated from the previous values, but editable if needed, as well as the same amount of attribute inputs, with the attributes types pre selected/populated with only the value of those attributes changeable. ***NOTE: all product items (where more than one exists) should have the same number, and same types of attributes. Ie, you dont want one product item allowing a size and colour, but then the next item allowing size, colour and material. This is illogical.***

The "add another product variation" button is always visible, and can be added multiple times to cover all variations.

The product item image would be selected from this info box as well. If you have already added images to the listing gallery, you can select from that gallery, or upload a new one in place which will get added to the gallery, and assigned to the product item.

If there are more than one product items boxes, we may want to either enfoce that the first product item box is the default item, ie, where the price comes from in all the places a product's price is displayed when searched/displayed in category listings.etc, OR, we also include a "is default product" toggle on each of the boxes, so they can switch betwen which one gets the default spot. Theoretically, you'd want the one with the lowest price as the default item, but the default item also dictates what IMAGE is displayed in listings where the product appears, so do we allow them to swap between their items, so they can rotate their default image/product, or be strict in this area? Open for discussion.

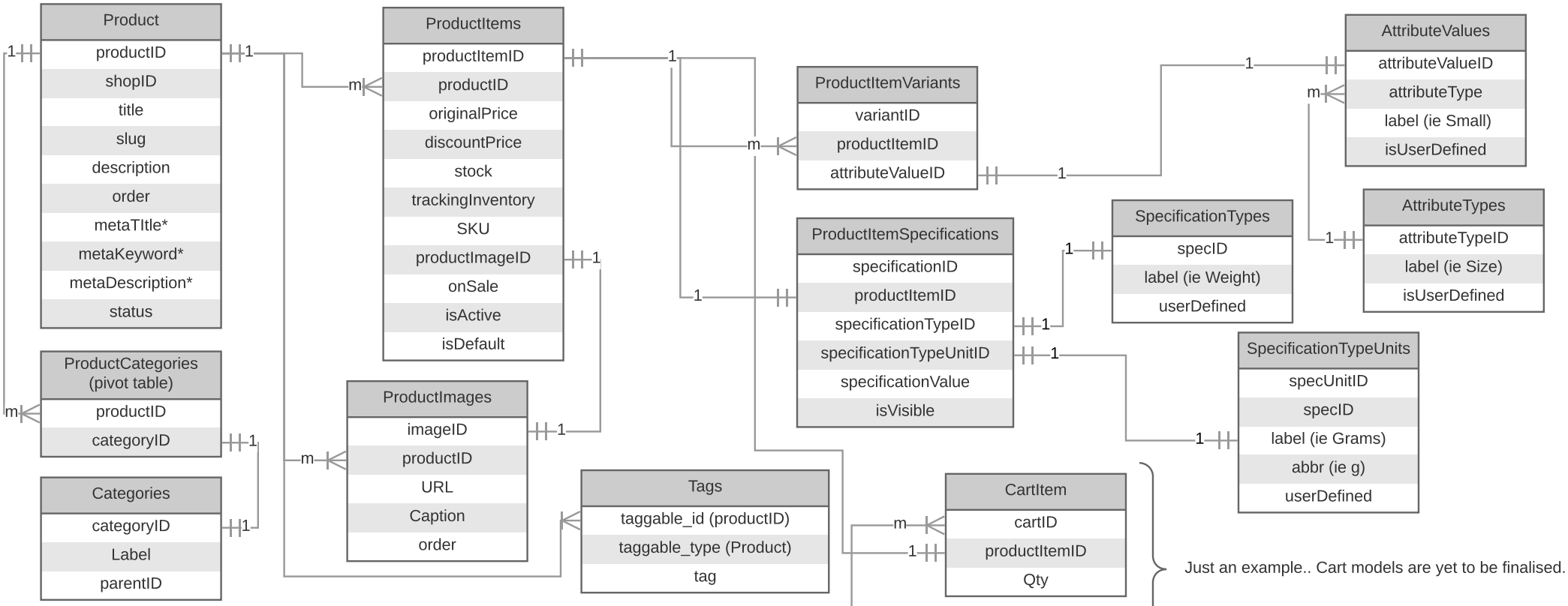
In short, with this solution, sellers who have products that dont have attribute variations, wont have to see anything relating to attributes, and those with attribute variations, can select to expand their product item to be configured with additional info, and also add more product items with different variations.

Note, if the default item has attributes defined, then they must have at least one other products item with a different set of attributes.
Why? If they have one product item, with attributes of Size - Small, and Color - Pink, but don't add any more product item variations/attribute sets, then on the product listing page there will be 2 dropdowns labeled, Size, and Color, with only one selectable value in each, Small and Pink.
In this case, it would be better to just name the product title - Small Pink T-shirt, and not use attributes at all, because there are no variations to select, and they're complicating the data for no reason

SEE MOCK UP PDF FOR "ADD PRODUCT" LAYOUT EXAMPLE FOR THIS SOLUTION

PRODUCT ER MODEL AND RELATED ER MODELS

Nicole Jay | March 29, 2018



Listing page - where the data comes from and how to handle it

Title - From the product model
Description - From the product model
Default Price - From the product's default product item, ie where isDefault = true
Main Image From the product's default product item, ie where isDefault = true

Product Options - Only relevant where the product has variations - product/items model
There will be a dropdown for each attribute type. Eg, Size | Colour. Each Dropdown will have distinct values constructed by each product item.

On page load, these will be pre-selected with the default product item's attributes. Also on page load, if there is more than one attribute type, ie Size & Colour, then the second attribute type's values will need to account for the scenario where the first type doesn't have a quantity for the second type's value.
Eg, T shirt's default item's Size is Small, but does not have a record or has 0 qty for the Colour - PINK, so Pink should be greyed out and should not be selectable when viewing the Colour dropdown after page load. Then, when the first attribute type's value is changed in the dropdown, Ie, to Medium, then the second dropdown should refresh with available colour options for the size Medium.

On select of final attribute combination
The Price will update with the selected item's price, and main image should be updated if the item selected has an image associated with it, the specifications table will also be updated with the items specification values. In some cases these may not change, but in some cases they definitely will.

Endpoints will be available to retrieve data models for all of these listing requirements

Note how only the productItem info is relevant in the scope of the cart

Simplified examples of retrieving values in PHP

- loop through $cart->cartItems as $cartItem -

To display product title in the cart
$cartItem->productItem->product->title

To display product item variations if applicable
- loop through
$cartItem->productItem->variants as
$productItemVariant -
$productItemVariant->value->type->label -
$productItemVariant->value->label

To display qty
$cartItem->qty

To display cost
$productItem->price * $cartItem->qty