**Tru-Medical: Saturday/Sunday 12/10-11/2011**



**footer on all pages** ( in application.html.erb )

**catelog request:**

1. instead of having category request create a new user, have it create a new catelog\_request record. It should also check to see if this is a user we already know about, and if so, link the catelog request record to that user, assocation called 'catelog\_requests'?

2. what happens after creating a new catelog\_request? right now it is trying to render views/users/show.html. Want it to give the user a quick confirmation that his request has been recieved, and will be mailed shortly. implies sending the form in using ajax; putting confirmation msg in the popup, and then having the popup close, or saying 'To close and return to ... click x"

3. expand the form to include a choice: does he want printed copy/copies of the catelog mailed to him, or would he like a pdf copy emailed to him? Maybe a 'click here to download a pdf copy' / fill in the form below if you would like us to mail you a physical copy: in which case we'll need his physical mailing address — soething we are not capturing now.

4. try other form builders: formtastic, its newer cousin: see RBates

5. check to see if anyone else thinks its odd to have to create a dummy @user in order to pass it to the form\_for when the form is going to be used to create an instance of the type.

**search for Sammons part number equivalent**

1. How going to do this?

2. Once you find it what? : go to page for the equivalent part in TruMedical catelog? What if the match is not exact?

**header on all pages (actually fields on the upper right )**

1. register/login: build own using Authlogic Rails plugin (or whatever used on aaox); want stuff to come up in popups above the home page, not go off to separate pages — and especially not the slow netsuite-generated ones.

**2nd, 3rd level category navigation**

1. update the category\_tree\_v3.xls file to add (i) the new images, (ii) the 3rd level categories. [DONE] 12/10

NOTE: Although this is a tree with a maximum depth of three, it does not go uniformly down to level 3 in all branches. Many go only to level-2. Force it to 3 by adding a dummy 3rd level 'All' category ito each 2nd level node that does not have 3rd level chldren as did in VMS? No, in that case, it was only a very small % of the 2nd level categories that required a dummy third level category 'All'. Here, well over half of the categories bottom out at level 2. Time for n-level category tree software. Will need both a nav-HTML builder and menu ( and/or category landing pages ). In this case I've hand-built the left-nav so the .rb script that builds the HTML for the left-name won't be necessary.

2. Make sure that we have all of the 'cateogry images' referenced in the excel file. Make sure they have been resized (to 120x80 jpgs), and are in the correct directory, eg.., /app/images/categories [DONE] 12/10

3. Reload the categories table in the database so that the categories pages will pull in these images. [DONE] 12/10

4. Restart the application and dbms servers, bring up a browser against the local version of the application, see how the category landing pages look. [DONE]

5. Fix anything that is breaking —

— Modalities

references to:

710025-000 (Accessories ) [ image file missing

710016-000 (Combination) [ image file missing ]

710054-00 (Electrodes) [ short a digit in the filename ]

710065-000 (Lotions and Gels) [ image file missing ]

problem: /public/images/categories contains only the thumbnails made from the -XXX files Bill sent; it does not contain the 710016-000 files.

fix: Find them in /public/images/[products, and make 120x80 thumbnails from them. Place the thumbnails in /public/images/categories. Do this for /images/products/modalities for now. Batch resize all of the files in that directory, copying them into /public/images/categories. Then go through /images/categories and delete all but the -XXX and -000 files.

problem: leaf-level category (modalities/clinical electrotherapy/ultrasound) is showing several images of e.g., Dynatron 125. Should it be? Yes: these are all of the individual products in this leaf-level category. Image of Dynatron Intellect Transport 710032-000 is missing. Why? Have 710031-00, 710033-000 and 710034-000 but not 710032-000. Why not?

6. insert the images for all of the top level product categories so that if Bill shows the Working Prototype to his partners, the left nav will work across the full product tree. This will also let us surface any 'missing-image' issues now, rather than later.

Within public/images/categories create a directory for each of the top level product categories. Put inside this directory a 120x80.jpg image of all products in that category, and of all of the composite images being used as sub-category images within this category. [ Where are the xxx's now? all in Modalities, or does Modalities contain only the ones that are really for subcategories of Modality. If the later, then batch convert the rest from the composite\_png files that Bill had Mandy send. Eventually go through them and pick out only the ones that are being used as category names, put those in /public/categories/all and adjust the program (views/categories/show.html.erb) to find them there. For now, add the subdirectory name into the pathname being used to find the category images. WARNING: this may not be possible with a category tree in which either the 2nd or 3rd level nodes can be leaf-level. If that proves to be a problem, copy them all in the /public/images/products/all. That was the case, since the category 'names' in the spreadsheet ( and therefore the database ) did not have underscrores, and the filenames did. ( And I didn't want to insert an obscure a regular expression to replace the spaces with underscores. ) However, images/categories/all only has the XXX images for modalities, not for the other top level categories. Where are they? Got them; have inserted the thumbnails that are used in the category landing pages in in /views/categories/alls. [DONE] 12/9

Go back and recreate all of the 120x80 thumbnails, shrinking the 169x179 jpgs (or the original pngs to 80 vertical, and then growing the canvas to 120x80, and adding a white background. Then sve the file to /images/cartegories/<correct category> or just /images/categories/all if don't want to keep a copy in the directories also.

— modalities [DONE] 12/10

— clinical electrotheraphy (6 images: 7100xx-000 where xx=16/25/31/33/54/65) [DONE] 12/10

— [ etc ]

— clinical supplies

— tables and traction

— fitness productds

— orthopedic products

— AT/Taping

— Evaluation

— Dailing LIving Aids

— Lymphedema

— Wound Care

7. insert images for all of the products

within leaf-level category pages

Shrink the roughly rectangular PNGs to 80 high wide and whatever width setting the 'constrain to same height/width ratio' yields. Then place that in the center of a canvas 120x80px in size, and set the canvas background color to white. Export as a jpg image, storing the result in a new directory: /public/images/products/thumbnails.

modalities

clinical electrotherapy

ultrasound [DONE] 10/12

stim

combination

for product\_types and products

Right now they are in /images/products/<top\_category\_name>. They are all 169x179 jpgs converted from the larger pns is /TruMedical/Content/.PNG Images all categories from Mandy Aug4 Sept27. ~706 of them in total across all categories. Normalize to rectangle in golden-mean ratio — or a rounded-to-the closest 10 pixel equivalent. Check GettyImages standard ratios.

Have leaf-level category pages find them within their subdirectories by adding the subcategory name to the image\_filename? means first substituting underscores for spaces in [sbu] category name.

7. Within the partials:

views/home/\_subcategories.html.erb

views/categories/\_leafCategories.html.erb, and

views/leaf\_categories/\_products.html.erb

— allow user to click anywhere on the subcategory block (image OR text) to get to the next level of subcategory or product.

— same for links to product pages from leaf-level category pages: clicking anywhere on the block should take you to the product page, not just on the live text link.

— highlight the text link on hover of the entire block (not just the text link); use the site orange rather than red that is currently in there; No: replace it with turning on the orange background for the entire blocks as doing on level-2 category pages.

8. the partial \_products.html.erb currently assumes that it will find the product images in images/products/modalities: generalize this by point it at images/products/all\_small and moving images on 120x80 (120wide,80 high) white pads into /images/products/thumbnails.

9. The show action in the LeafCategoriesController makes a similar test assumption: it always returns in @products the set of products from category 9. It needs to accept the category number from an argument on the call. Thta argument needs to be set up in the link\_to macros in views/categories/show.html.erb.

10. sort [sub]categories within category so that things are shown on the page in the order specified in the category-tree entries.

11. get left-nav partial to work on category pages as well as home page.

12. remove the sibling nav: just confusing; Instead make the entries in the bread crumb trail live so that a user who wants to see other altneratives can go back up the tree to the [sub]category landing page that got him to where he is now.

13. left nav currently closes the fly-out panel for one menu item when you hover over another. That doens't work for the last one, if you, e.g., move your mouse from payor to either of the blocks in the rectanagle with Tru-Value products; in that case, since you haven't moused over another menu option, the 'payor' overlay remains up. Could we add a special-case mouse leave event to just the last item in the menu list without screwing up the browser's event stack state? If not, how about explicitly flushing the browser event queue? The API is probably unique to each different browser. Alternatively, put the code to close existing menu overlays into the event handlers for mousing over Promotions or Tru-Value Products. Clearer — and need to define what Bill wants to happen when the user does mouse over these elements anyway. One potential gottcha on this tack: Promotions and True-Value Products are bulit in on partial, the menu in another. Would need browser state to be visible across the two. Should be Ok.

**product pages**

1. insert a simple product page, so that Bill can walk down the tree and get to the bottom. Make sure we can handle branches of the product tree that are only 2 levels deep as well as those that go down the full 3 levels; this is a navigation issue, not something for the product page.

2. Note: the product images in images/products/all\_small and all\_medium DO NOT include all fo the products. I could not even find the ten images for modalities/electrotherapy/ultasound — unless for the product 7102490-002 and 710249-005 and 710249-010 we always use the image with name 7100249-000.jpg ( which is what the \_product partial does not ). WE ARE MISSING one image for our test subtree: 710032-000.png

3. insert working buttons to add-to-cart and checkout so that we can test the full path to ordering a product or set of products ( before we necessarily have the product group pages working as in the catelog. If we cleaned up the scrollers on the home page, in inserted the search, the sammon-part-number search, and made the catelog request form create [user]<-->>Catelog\_Request records so it would work with existing users rather than always assuming that they are new, we would have a site that Bill could put up, deciding the add the product group pages in a subsequent point release. [ Footnote: we would have to get the matrix item support working so we could handle colors and sizes in any such release 1.0. ]

**product group pages**

1. reread my notes

2. review the design of the database classes/tables: reinsert the 4th table: line, type, model, configured\_product(\*), rather than trying to collapse it to 2? If insert Rails-side Admin subsystem then modeling the product class hierarchy completely is not much harder to administer than trying to keep it brief enough that we can push it into the Netsuite 'Item' table in an 'encoded' form. The downside of the encoded-in-the-Item table approach is that the site administrator would always have to remember why it was there, and how to work with it. Would also add all four the record types to Netsuite, but if we put the Admin subsystem on the Rails side, then we really wouldn't need anything on the Netsuite side other than the single predefined 'Item' record type with the built-in 'matrix' option enabled for the products that come in color/size combinations — essentially the 'configured products' that are actually being ordered, shipped, and tracked in the Netsuite financials.

3. create them with migrations

4. load them with data

5. write the HTML/CSS/Ruby for the pages/overlays involved.

(\*) How about Line, Product, Model where:

Manufacturer <->>ProductLine

ProductLine<->>Product

Product<->>Model

If a product has models, then it is Models that correspond to Netsuite Items

If it does not, then it is Products that correspond to netsuite Items.

Note that some products may come in sizes, and colors. This is handled by defining 'matrix items' on the Item in netsuite, one for size, one for color, and passing in orders for a particular Product (=Netsuite Item) that comes in a particular size and/or color.

Accessories are defined on Products ( not Models ) — in the case of the Intelec Transport Ultrasound unit, the unit is the Product; it comes in 4 models depending on the size of the applicator, and it has as accessories: cart, battery pack, carry bag, and any of the independently orderable applicators.

The only odd bit about this, is that the extract of items from Netsuite would be loaded into what? the Model table? And lots of Products would have a single model? The other alternatives is to use the word Product for what is here Model, and then introduce Product\_Type for what was here Product. Then netsuite items ( or at least that those don't have matrix parameters) map to Products. And we have to invent a term for the combination of an item in a given size and color. I suppose we could refer to that as a Product in a given size and color, or a Product Configuration. The problem is that English tends to use the simple term ( in this case 'Product') for the thing you think of most immediately, and qualifiers for less common things. And in this product set, when you are thinking about the simpler ones, model#2 fits this bill better. When you are thinking about complex ones, model#1 fits this bill better. I'm going to go with model #2 because it fits 80% of Tru-Medical's product line, and accept the oddness of having Product\_Types in the complex case.

So for the complicated case we have:

Manufacturer <->> ProductLine

ProductLIne <— contains —>>ProductType

ProductType <— has—>> Product

Product <— comes in —>> [ Sizes, Colors ]

and

Category <—>> Category, and we distinguish two subtypes: top-level and leaf-level categories

leaf-level Category <—>> Product\_Type

or?

leaf-level Category <—>> Product

We could hand this by introducing product types even for the products that don't have them ( that is, in the language of model#2, products that don't have models )

Or (option B) we could do it by defining on the Category record, both the associations product\_types and products. Then in the leaf-level category show.html.erb page, if the product\_types association is empty for the category, we list all of the products. Otherwise (if the product\_types assocation IS populated) we list the product\_types. And the product\_type show.html.erb page will have on it all of the products and accessories that go with this product\_type. Or, to try to bridge the two sets of terminology, we might say that a product\_type has ( or 'comes in' ) one or more models, each of which is referred to by the inventory and financial systems as a separate 'product'.

**Ok, so we're going to go with model #1 (ProductType <—>> Product ) and Option B for mapping leaf level categories to ProductTypes and Products.**

Mechanics of implementing it, from where the software stands now:

1. We already have the models: Category, ProductType and Product.

2. Category already has a has\_many :products relationship defined on it; add a has\_many: product\_types relationship. [DONE] 12/10

3. ProductType already has a has\_many :products relationship defined on it. Add a belongs\_to: category relationship. [DONE] 12/10

4. Product has a belongs\_to: product\_type. Add a belongs\_to: category. [DONE] 12/10

5. For the complex case, add classes Manufacturer and ProductLine. Each has a name, and a logomark. ProductLine has a description also. link ProductLine <->> ProductType. [DONE] 12/10

6. Create/apply a migration to add foreign key for product\_type ( in addition to the existing one for leaf-level category) to products. Add column for this to the excel sheet from which the products table is being loaded; and reload the database [ populate this column and reload the database again when get to building/testing the complex case below ] [DONE] 12/10 used products\_all\_wModels\_1018.xls as base.

7. Modify views/leaf\_categories/show.html to see whether it should be listing product\_types or products, and call the appropriate partial. How? For now, put a column has\_product\_types? in the Category table. leaf\_category/show.html.erb line 216ff. [ Test the if ] [ in Modalities/Electrotheraply, chose Accessories or Lotions and Gets. Nether has the type/model distinction and should therefore render the partial 'products'. Does it?

8. A partial for Products exists. Create a new partial for Product Types. [DONE] 12/10

9. There is a views/product\_types/show.html page. Check it when get to step 13.

10. Build a views/products/show.html.erb page [DONE] 12/10 — get it to compile; get it to work: may need step 12 before can test it; need a path of pictures through the tree of landing pages to get here.

problem: the category\_id' foreign key in the file used to load the products table is not filled in, so we don't know have the llc<-—> products mapping we need.

solution: Build a ruby script that can be run in the Rails console to fill this field in in the database records after the products spreadsheet is loaded, to wit:

Product.all.each do |p|

p.category\_id = Category.where( :name => p.category\_l3).first

p.save

end

To avoid problems with l3 category names that occur multiple times in the categories table (e.g., 'Accessories' ) will have to do this more rigorously eventually, but this will serve for the testing of the product and product\_type pages I am trying to do now. It will return the correct Accessories for the initial product subcategory: Modalities/Clinical Electrotherapy.

11. Create a products controller: products\_controller.rb, and put a show action in it. [DONE] 12/10

12. Put in the pictures to get us down to an example of a simple product ( no models, no sizes, no colors).

— pick one:

— test the products page built in step 7 above: confirm that you can

(i) get to it from leaf-level category page, and

(ii) push orders from it into the standard Netsuite shopping cart

(iii) create a more subtle acknowledgement that you have put an order in the cart

(iv) checkout and have the order flow correctly through to the Netsuite backend. This means that we will need skins for netsuite generated pages. What should they look like?

13. Put in the pictures to allow a site user to get to an example of a product that comes in sizes and/or colors:

— pick one:

— test the products page built in step 7 above: confirm that you can

(i) get to it from leaf-level category page,

(ii) add code to the views/products/show.html.erb page that lists the size/color options and lets the user select which he wants.

(ii) push orders from it into the standard Netsuite shopping cart, and

14. hand build the database for the electrotherapy example of the complex case. See that you can get to the product\_type page, and order products of this type, accessories, etc direclty from that page (without going to a 'product' page.

[

the following extracted from comment section at the head of views/product\_types/show.html.erb

# test 1

# Get record for Intelect Transport Electrotherapy unit

# This Product has a single Model: 710031-001, a 2-channel unit

# test 2

# Get record for Intellect Transport Ultrasound

# This Product has four Models:

# 710032-001 w/1cm applicator

# 710032-002 w/2cm applicator

# 710032-005 w/5cm applicator

# 710032-010 w/10cm applicator

# It also has a set of Accessories — some of them (cart, battery pack, carry bag) are the same

# as those for the Intelect Transport Electrotherapy and Combo units; others are unique

# to the Ultrasound unit — the 4 different applicator sizes; if user buys the unit with one

# applicator size, he can also buy applicators in other sizes.

# Note that the applicators are Accessories to both the Ultrasound unit and the Combo Unit.

# For now, just repeat all of them as Accessories on the Product. Don't try to model

# accessories for the Line.

# To make it concrete, in the example we have been using, each of these terms (Line, Product, Model, Accessory) would refer to the following:

# ProductLine -> Intelec Transport

# Product —>> Intelec Transport Electrotheraphy, IT Ultrasound, IT Combo Unit

# Model —>> for IT Ultrasound: 710032-001 through 71-0032-010: one for each of the different applicator sizes

# Products would then have Accessories.

# How would the category tree work with this?

# Entries in the leaf-level category page would point to Products,

# e.g., the 'ultrasound' entry in the leaf-level category page modalities/electrotherapy/ultrasound would

# contain references to ultrasound products from different manufacturers and/or different lines made by

# the same manufacturer ( in chattanooga's case: ultrasound units from the three lines:

# Intelec Legend, Intelec Transport, and Intelec Legend XT.

]