nextMonet™

# NextMonet Framing Project Functional Specifications

Jack Hodges

January 19, 2001

## Abstract

This document serves to define the basic market and user requirements for the tools that will allow NextMonet customers to view potential art purchases in a frame and to receive a quote on that frame. This project will consist of one "major tool" that will have a high-level set of functionality and several functional "mechanisms" that will support the implementation for sales, shipping and receiving. The model incorporates a consistent look and feel with the existing site.

The requirements for the major tools will be specified by the combination of the functional use cases for the framing model, the user interface (UI) elements, the story-board of pages that will implement the use cases pertinent to the framing tool, the object model, and the functional mechanisms in the tool.

### Objective

The framing project is intended to provide content usability support by providing the user with a mechanism for framing art for potential purchase, and thus allowing the user to see how the art might appear once framed. The targeted art is anything that can be framed, such as a print, photograph, or painting.

The user will be able to select a 'frameit'-like button on the Working Page (below the artwork image), if the object is frameable, that will take him/her to a new page which is similar in UI format to the Working Page. The artwork in frame will be viewable on the left, with controls and information on the right. The right hand side will be tabbed like the Working Page. A new button under the framed image will take the user directly to a 'View in Room' feature or to the Order page.

The framing page should come up with the recommended framing options, detailed on the information half of the page. This will require revisiting the entire database to classify every frameable image.

### Tools

The major tool that will be incorporated in to the site described listed here.

1) **Matte Selection Tool**: This tool/control allows the user to select the matte type and the picture will be mounted on. There will only be one matte color offered for display online.

2) **Moulding Selection Tool**: This tool/control allows the user to select the type/color of the frame moulding the picture will be mounted in. Photographs will only be displayed with the black metal frame, and prints/watercolors will only use the wood frames.

### Mechanisms

1) **Framing Quotation Mechanism**: This mechanism takes the selected or automatic options and calculates the associated framing quote.

2) **Framing Display Mechanism**: This mechanism takes the selected options, constructs the layered image and background colors, and displays them on a page.

3) **Framing Sales Mechanism**: This mechanism takes the selected options and adds them to an order if the user chooses to purchase the piece with the displayed framing.

## Constraints

There are seven constraint types known to exist at the onset of this project that will affect the design of the tools and mechanisms: (1) mat type, (2) moulding type, (3) image display, (4) framability, (5) shopping cart display, and (6) framing calculation. The constraints will be described below after a brief outline of affected logics.

### Logic

There is logic that must be applied to the framing process that is already known, as outlined below:

- There should be a 'framing' button on the 'working page' and there should be some kind of 'return' button on the 'framing' page that returns to the 'working page'.

- There will be a framing calculation on the 'framing page'.

- There will be a 'default' frame recommendation that is initially displayed, along with the specifics (frame type, mat type) that goes along with it.

- Objects which aren't 'framable' will not be offered as framable, meaning that the 'framing' button will not appear. This will also apply to objects that are already framed. This information can be found in the 'framing' field of the 'art_objects' table.

- The mat will be displayed to the physical dimensions of the paper in cases of photographs, prints, and watercolors.

### Mat Color

NextMonet only provides a mat in white and offwhite. Thus there is no need to offer the user the choice of mat colors. Instead, a color between these two colors will be chosen and all mat will be this color.

### Mat Type

There are only two mat types: floating and overmounted. Mats can only be used on prints, works on paper, watercolors, and photographs. The user will be able to select the mat type as long as a mat can be used on the artwork. Both mat types will assume a specific size with respect to the distance to the visible portion of the artwork. That is, the user will not have a selection of mat sizes: this is determined by the framer to optimize the artwork viewability, and the display model will present one possibility. A caveat will be required so that the viewer understands that the finished piece could have a mat of different size.

### Moulding Type

There are two moulding types available: wood and metal. There are six wood mouldings available: (1) maple clear, (2) maple white, (3) maple black, (4) walnut, (5) silver, and (6) gold. There is only one, black, metal moulding, and it can only be used for photograms. The user will not have a selection of mat or frame sizes: these are determined by the framer to optimize the artwork viewability. The user will not have a selection of frame sizes: this is determined by the framer to optimize the artwork viewability.

## Framing Display

The framed image must be proportional to the mat and frame on the webpage. There are two issues in making this happen. First, the only image that is scanned and compared to its actual size is the image. The other pixel sizes, for the frame and mat, are scanned or otherwise gotten into jpeg format, but not compared to their physical dimensions. So a relatively accurate pixel size must be obtained for the frame and mat. This means that the actual size of the work must be known and not just the viewable size. At the time of the start of this project, physical dimensions of the artwork were stored in the Embark CMS, but not being migrated to the Oracle database. This has since changed. The visible dimensions of the artwork (in inches) are represented as height and width. The converted dimensions of the artwork (in pixels) are represented as pixel_height and pixel_width. The physical dimensions of the artwork (in inches) are represented as image_height and image_width. The ratio: pixel_height/height should be the same as the ratio: pixel_width/width. We can thus calculate the pixel size of the physical dimensions as follows:

pFH = fH x (pixel_height/height)

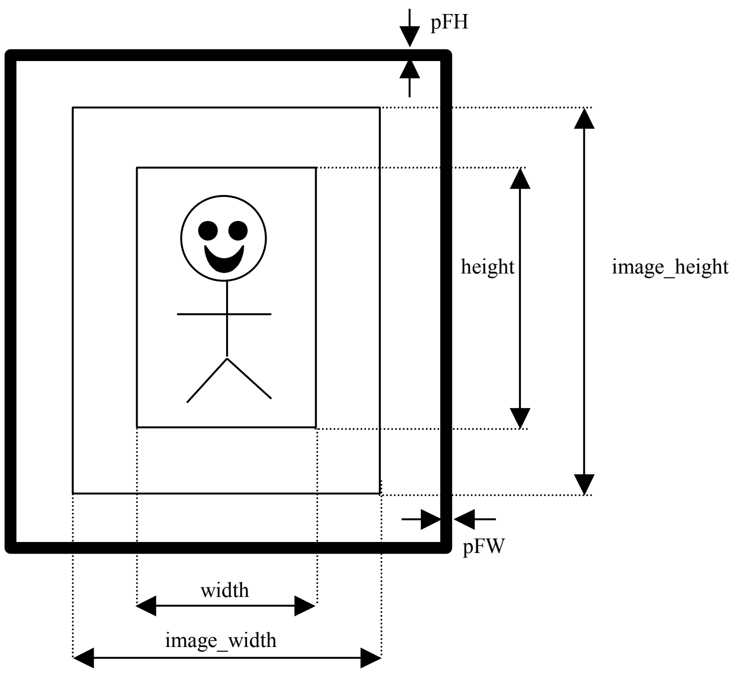
pFW = fW x (pixel_width/width)

pMH = mH x (pixel_height/height)

pMW = mW x (pixel_width/width)

pimage_height = image_height x (pixel_height/height)

pimage_width = image_width x (pixel_width/width)



where pFH and pFW are the pixel frame sizes, fH and fW are the actual frame sizes (in inches), pMH and pMW are the pixel mat sizes, mH and mW are the actual mat sizes (in inches), pimage_height and pimage_width are pixel physical dimension sizes, and image_height and image_width are the actual physical dimensions (in inches). Since all of the physical dimensions are known, and since the pixel ratios are known, the new pixel sizes are known.

The problem with this set of assumptions is that we have no way of knowing whether the pixel_height/height ratio really applies to the frame or mat.

## Frame Size

In order to produce a reasonably-appearing framed image on the website, a balanced frame width should be provided. Given that we are only providing a single set of images for each frame type, the frame must be scaled properly to the artwork prior to placement. The frame width in reality is selected by the framer to match the artwork as best as possible. Having measured all of the framed artworks available at NextMonet at the time, and having asked all of the in-house experts their opinions on the frame sizes on these artworks, I have come to the conclusion that there are artwork height to frame thickness ratios above which the frame will appear too thin, and below which the frame thickness will appear too thick. The range is as follows:

22 < frame_ratio < 72

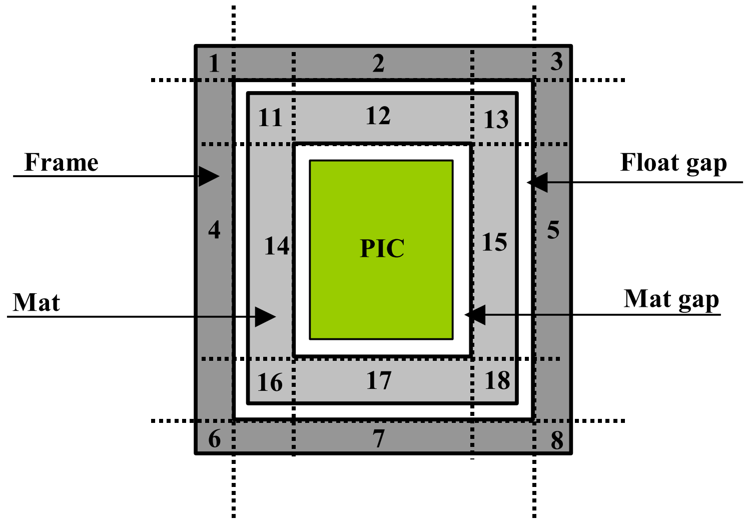
This is quite a spread. Moreover, the darker the frame, the easier it is to accept a thin frame on a large art piece. Thus, I recommend that we use the following fixed frame_ratios for the website:

| | |
|---|---|
| 25 | white |
| 35 | silver |
| 45 | gold, clear |
| 55 | mahogany |
| 65 | black |

With these ratios, once a total height/width is calculated, the frame size can be calculated and the positioning algorithms can be executed.

## Mat Size and Placement

Second, the mat and frame images are broken into eight pieces as shown below:



1-8 - Frame images
11-18 - Mat images

The cuts were made at these locations so that the frame wood and metal grains would match up and so that the web sections (2, 7) and riser sections (4, 5) could be scaled in width and length, respectively, without adversely affecting the scale of the other sections. The mat images were broken up the same way so that the same placement algorithms could be used on them. When these sections are placed onto the webpage, they must be placed based on where the upper lefthand corner of the image is, since that is the reference for the entire display. This requires us to know what the size of each of these sections is, and then to move that section into place.

One issue that has arisen is that many pieces of art do not have image_height and image_width dimensions in Embark. For now it is being assumed that null fields in the database for these fields will be interpreted to mean that they are the same as the height and width dimensions, respectively.

The mat gap shown in the figure is a fixed size of ¾ inch, converted to pixels, that is applied to works on paper, watercolors, prints, and possibly photographs. The float gap is used when the work is mounted as a float, in which case the mat is under the piece and is represented entirely by the float gap. In the case of the float mount, the artwork overlaps the mat by an amount that makes up the float gap. The float gap will be fixed at 2 inches. The mat size can be fixed in all cases where the following relation holds:

- **Over Mount**

  image_height <= height + 2 * mat_size + 2 * mat_gap

  image_width <= width + 2 * mat_size + 2 * mat_gap

In those cases, the total_height and total_width equal the right hand sides of the relations above. In the case where the above relation fails, the mat_size is increased to the larger of the following:

  mat_size = (image_height – height – 2 * mat_gap) / 2

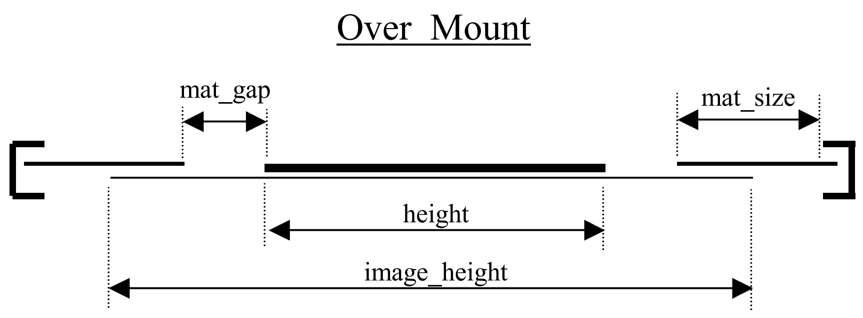
  mat_size = (image_width – width – 2 * mat_gap) / 2

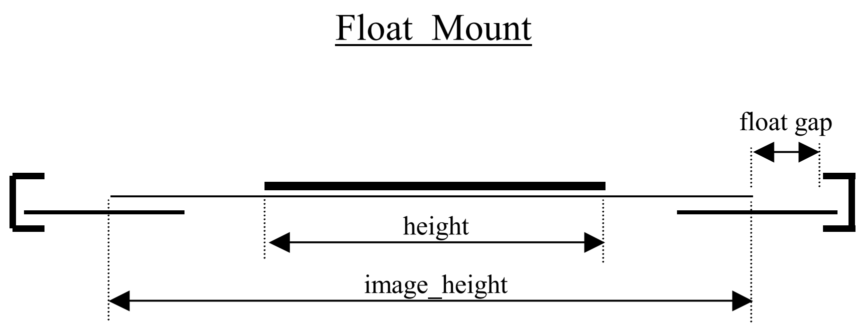
That is, the total_height = image_height, and total_width = image_width, and the mat takes up the space between the image and the frame.

- **Float Mount**

  total_height = image_height + 2 * float_gap

  total_width = image_width + 2 * float_gap

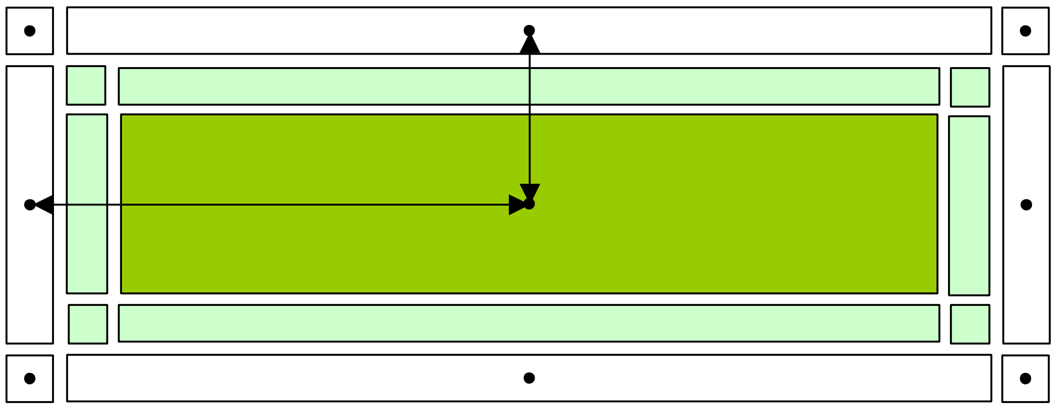


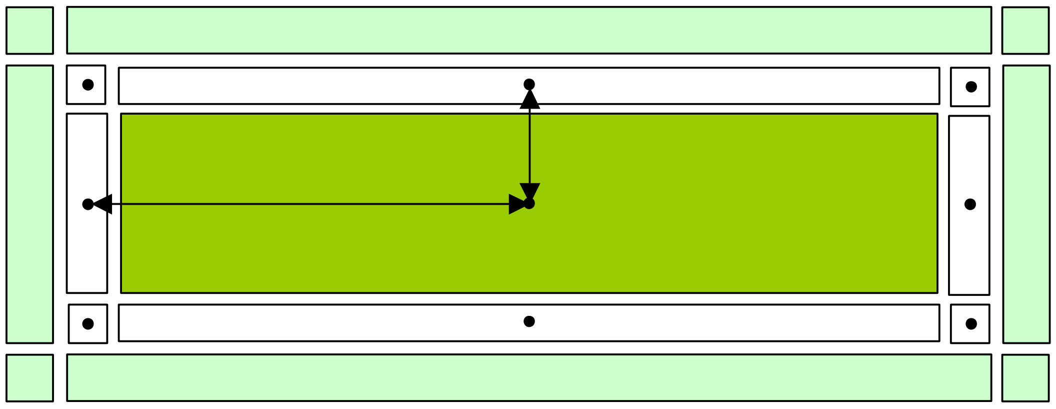


It should be noted that the same mat images will be used in both overmount and float mount cases, but that in the latter the bevel will not be seen.

## Placement Algorithms

The algorithm for placing components requires that the centroid of the component be known, along with the height and width of the component. The following figure illustrates the means for determining the placement algorithms:





The placement algorithms are all based on the centroid of the art_image, as depicted above. The centroid will be referred to as artX, arty. The following are the algorithms for the over mount frame component positions, assuming that the frame has been scaled and that the image_height and width allow for the fixed mat size:

## Overmount Normal

```
ulFrameX = artX – width/2 – mat_gap – mat_size – ulFrameWidth/2
ulFrameY = artY – height/2 – mat_gap – mat_size – ulFrameHeight/2

lFrameX = artX – width/2 – mat_gap – mat_size – ulFrameWidth/2
lFrameY = artY

llFrameX = artX – width/2 – mat_gap – mat_size – ulFrameWidth/2
llFrameY = artY + height/2 + mat_gap + mat_size + ulFrameHeight/2

uFrameX = artX
uFrameY = artY – height/2 – mat_gap – mat_size – ulFrameHeight/2

urFrameX = artX + width/2 + mat_gap + mat_size + ulFrameWidth/2
urFrameY = artY – height/2 – mat_gap – mat_size – ulFrameHeight/2
```

```
rFrameX = artX + width/2 + mat_gap + mat_size + ulFrameWidth/2
rFrameY = artY

lrFrameX = artX + width/2 + mat_gap + mat_size + ulFrameWidth/2
lrFrameY = artY + height/2 + mat_gap + mat_size + ulFrameHeight/2

lFrameX = artX
lFrameY = artY + height/2 + mat_gap + mat_size + ulFrameHeight/2
```

These algorithms are only modified by removing the last term and halving the mat_size terms in order to apply to the mat figure placements.

## Overmount Large Paper

```
ulFrameX = artX – image_width/2 – ulFrameWidth/2
ulFrameY = artY – image_height/2 – ulFrameHeight/2

lFrameX = artX – image_width/2 – ulFrameWidth/2
lFrameY = artY

llFrameX = artX – image_width/2 – ulFrameWidth/2
llFrameY = artY + image_height/2 + ulFrameHeight/2

uFrameX = artX
uFrameY = artY – image_height/2 – ulFrameHeight/2

urFrameX = artX + image_width/2 + ulFrameWidth/2
urFrameY = artY – image_height/2 – ulFrameHeight/2

rFrameX = artX + image_width/2 + ulFrameWidth/2
rFrameY = artY

lrFrameX = artX + image_width/2 + ulFrameWidth/2
lrFrameY = artY + image_height/2 + ulFrameHeight/2

lFrameX = artX
lFrameY = artY + image_height/2 + ulFrameHeight/2
```

## Float Mount

```
ulFrameX = artX – image_width/2 – float_gap - ulFrameWidth/2
ulFrameY = artY – image_height/2 – float_gap - ulFrameHeight/2

lFrameX = artX – image_width/2 – float_gap - ulFrameWidth/2
lFrameY = artY

llFrameX = artX – image_width/2 – float_gap - ulFrameWidth/2
llFrameY = artY + image_height/2 + float_gap + ulFrameHeight/2

uFrameX = artX
uFrameY = artY – image_height/2 – float_gap - ulFrameHeight/2
```

```
urFrameX = artX + image_width/2 + float_gap + ulFrameWidth/2
urFrameY = artY – image_height/2 – float_gap - ulFrameHeight/2

rFrameX = artX + image_width/2 + float_gap + ulFrameWidth/2
rFrameY = artY

lrFrameX = artX + image_width/2 + float_gap + ulFrameWidth/2
lrFrameY = artY + image_height/2 + float_gap + ulFrameHeight/2

lFrameX = artX
lFrameY = artY + image_height/2 + float_gap + ulFrameHeight/2
```

### Framability

The Oracle art_objects table has a field called `framing` which can have a String value of: “null,” “Framed,” “Framable,” “Not.” If an art_object has an “Not” or “Framed” value, the framing button on the working page should not show up. Also, if the artwork has a depth dimension, it is a sculpture and cannot be framed. Such a piece would have a null `framing` field, so this should be checked first. If the art_object has a “Framable” or “null” value, it will be assumed that it can be framed and the framing button will show up on the working page.

Art_objects also have a String `short_description` field which tells whether it is a “canvas,” “paper,” “watercolor,” “photograph,” or “print.” If the value is canvas a mat cannot be used and the mat selection tool will be disabled. Photographs will only be allowed to select framing, and not the type of frame. All photographs will use overmounted mats. The `short_description` field is supposed to end in a prepositional phrase that looks like:

on paper
on canvas

but doesn’t always do so. As such, the model will apply the rules above to see if the piece is farmable, and will then use the following rules to do further constraint checking and calculate the framing price:

Rule1: If the piece has a depth field, it cannot be framed
Rule2: If the field has “on blah type” then logic/price for type
Rule3: If the field doesn’t have “on blah type”, then if the field has “type”, then logic/price for type

#### Artwork Types from Art_Objects Short_Description

The following are the types that are currently found in the art_objects short_description field in the Oracle database:

| Type | Category | Type | Category |
|---|---|---|---|
| Mixed media | 2 | Vibrachrome | 3 |
| Lithograph | 3 | Intaglio | 3 |
| Print | 3 | Photograph | 3 |
| Etching | 3 | Photogravure | 3 |
| Watercolor | 3 | Wood | 2 |
| Painting | 2 | Cibachrome | 3 |

| | | | |
|---|---|---|---|
| Plywood | 2 | Mahogany | 2 |
| Plaster | 2 | Cloth | 2 |
| Rubber | 2 | Duratrans | 3 |
| Printmaking | 3 | Redwood | 2 |
| Oak | 2 | Wire | 2 |
| Fiberglass | 2 | Teakwood | 2 |
| Bronze | 2 | Clay | 2 |
| Aluminum | 2 | Photogiclee | 3 |
| Linocut | 3 | Ceramic | 2 |
| Dirt | 2 | Drawing | 3 |
| Monotype | 3 | Oil | 2 |
| Embroidery | 2 | Weaving | 2 |
| C print | 3 | Marble | 2 |
| Travertine | 2 | Basswood | 2 |
| Bible | 2 | Nfs | 2 |
| Latex | 2 | Acrylic | 2 |
| Photogram | 3 | Pastel | 4 |
| Felt | 2 | Aquatint | 3 |
| Silkscreen | 3 | Porcelain | 2 |
| Stone | 2 | Slate | 2 |
| Spit-bite | 3 | Gouache | 3 |
| Ilfochrome | 3 | Snakeskin | 2 |
| Woodcut | 3 | Poster | 3 |
| Photolitho | 3 | | |

## Artwork On-Types from Art_Objects Short_Description

The following are the types that are currently found in the art_objects short_description field in the Oracle database that are referred to in a prepositional phrase “on type”:

| On-Type | Category | On-Type | Category |
|---|---|---|---|
| Plywood | 2 | Paper | 3 |
| Woodpanel | 2 | Wood | 2 |
| Canvas | 2 | Board | 2 |
| Topographical map | 3 | Mylar | 2 |
| Linen | 2 | Photograph | 3 |
| Clayboard | 2 | Plaster | 2 |
| Steel | 2 | Masonite | 2 |
| Litho sheets | 3 | Papyrus | 2 |
| Catalog | 2 | Parchment | 2 |
| Duratrans | 3 | Serigraph | 3 |
| Velvet | 2 | Resin | 2 |
| Aluminum | 2 | Maple | 2 |
| Game table | 2 | Sold | 2 |
| Newsprint | 3 | Gampi | 3 |
| Plexiglass | 2 | Book pages | 3 |
| Glass | 2 | Cardboard | 2 |
| Styrofoam | 2 | Cheesecloth | 2 |
| Embroidery | 2 | Foamcore | 2 |
| Acetate | 2 | Burlap | 2 |
| Graphite | 2 | Porcelain | 2 |

| | | | |
|---|---|---|---|
| Formica | 2 | Empty | 2 |
| Vinyl | 2 | Plexix | 2 |
| Vellum | 3 | NM Gift Cert | 2 |
| Rag | 2 | Bristol | 2 |
| Homosote | 2 | Stretcher | 2 |

## Shopping Cart Display

If the user selects to purchase the framed artwork, then it should be displayed framed on the shopping cart. At present, the shopping cart page uses the `ImageURLFactory.DisplayObject` and `ImageURLFactory.cellSize` methods to display the selected artwork in a small (75 pixel) size. These take the SKU and send it to the image server. When the same operation is done from the framing page, the last url sent to the image server should be used to forward to the shopping cart. That is, the URL should be written to the session object so that it will be persisted for the session.

## Framing Calculation

The framing calculation is based on the width and height of the frame. To calculate the price, add up the length and height of the piece to figure out the United Inches. For works on paper and pastels, you will need to add 8 inches to the United Inch total to account for mat. Below are the tables:

| United Inches | Works on Canvas | Floaters for Canvas | Works on Paper | Pastels |
|---|---|---|---|---|
| 0-20 | $87.07 | $142.60 | $118.17 | $134.82 |
| 21-25 | $99.22 | $169.00 | $133.07 | $149.72 |
| 26-30 | $111.37 | $195.40 | $151.72 | $168.37 |
| 31-36 | $125.17 | $223.53 | $174.65 | $192.08 |
| 37-42 | $141.38 | $252.48 | $209.02 | $227.58 |
| 43-52 | $160.93 | $293.03 | $259.00 | $278.02 |
| 53-60 | $189.52 | $341.37 | $300.55 | $323.48 |
| 61-66 | $202.50 | $367.77 | $324.33 | $347.27 |
| 67-72 | $220.08 | $399.00 | $354.60 | $379.70 |
| 73-78 | $249.43 | $431.28 | $440.28 | $468.05 |
| 79-85 | $265.88 | $459.13 | $475.25 | $503.65 |
| 86-90 | $287.52 | $493.78 | $516.62 | $548.75 |
| 91-100 | $315.53 | $548.32 | $557.47 | $590.37 |
| 101-108 | $354.78 | $614.67 | $724.68 | $763.68 |
| 109-114 | $382.55 | $655.77 | $770.98 | $809.60 |
| 115-120 | $396.37 | $682.17 | $872.78 | $918.38 |
| 121-126 | $421.78 | $720.78 | $915.85 | $966.95 |
| 127-135 | $442.93 | $753.60 | $947.68 | $1,001.58 |
| 136-144 | $560.58 | $901.18 | $1,037.80 | $1,134.55 |

For example, for a work on paper that is 16” x 20”, the United Inches would be 36”, and we add 8” for the mat, making 44”. Using the table, we see that the framing cost would be $259.

# Design

This section is intended to give describe the framing project design model. There are four components to the design:

- Use cases
- User interface elements
- Page flows (storyboards)
- Object model

## Use Cases

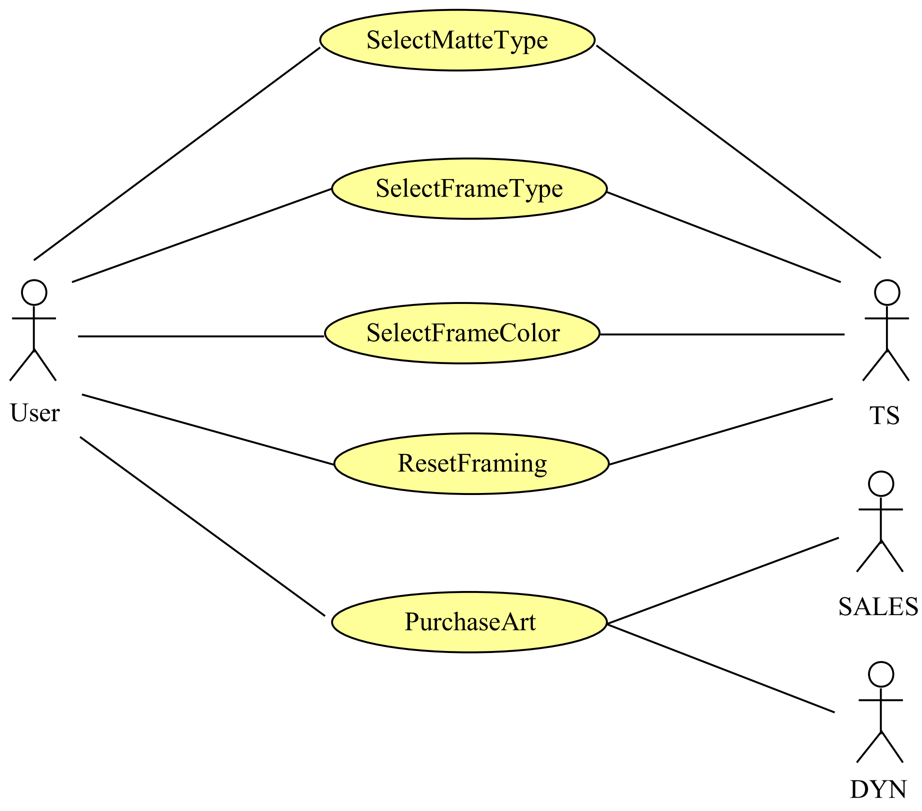
These talk about the use cases for each tool that is specified and the primary and major secondary scenarios fore each use case. First the actors for the project are defined, and then the use cases themselves.

## Actors

There are four actors identified by the use cases for the framing model:

- **USER:** Configuring/viewing the framed art
- **TS:** The TrueSpectra Image Server (layering model)
- **DYN:** The Dynamo Application Server
- **DB:** The database

## User-Driven Use Cases



## USER SelectMatType into TS

**Precondition**: An artwork is viewable on the working page and the artwork is frameable.

**Flow of Events**:

Primary Scenario

**1)** USER selects framing button from working page

**2)** USER manipulates mat type

**3)** TS displays artwork with selected mat type (with default color unless already specified)

**Postcondition**: The artwork is displayed with an overmat (default)

### USER SelectFrameType into TS

**Precondition**: An artwork is viewable on the working page and the artwork is frameable.

**Flow of Events**:

Primary Scenario

1) USER selects framing button from working page
2) USER manipulates moulding type
3) TS displays artwork with selected moulding type (default color unless one has been selected)

**Postcondition**: The artwork is displayed with the selected moulding type (default color unless already specified)

### USER SelectFrameColor into TS

**Precondition**: An artwork is viewable on the working page and the artwork is frameable.

**Flow of Events**:

Primary Scenario

1) USER selects framing button from working page
2) USER manipulates moulding color
3) TS displays artwork with selected moulding color (with default type unless one has been selected)

**Postcondition**: The artwork is displayed with the selected moulding color (with default moulding frame unless already specified)

### USER SelectReset into TS

**Precondition**: An artwork is viewable on the framing page in some degree of framing

**Flow of Events**:

Primary Scenario

1) USER selects reset button on framing page
2) TS clears all buffers and displays artwork on page

**Postcondition**: The artwork is displayed in same manner as on working page.

### USER SelectPurchaseArt into DYN

**Precondition:** An artwork is viewable on the framing page and a pricing quote has been displayed.

**Flow of Events:**

Primary Scenario

1) USER selects PurchaseArt button
2) DYN sends POST information framing to the order bean
3) DYN adds order bean information to shopping cart page

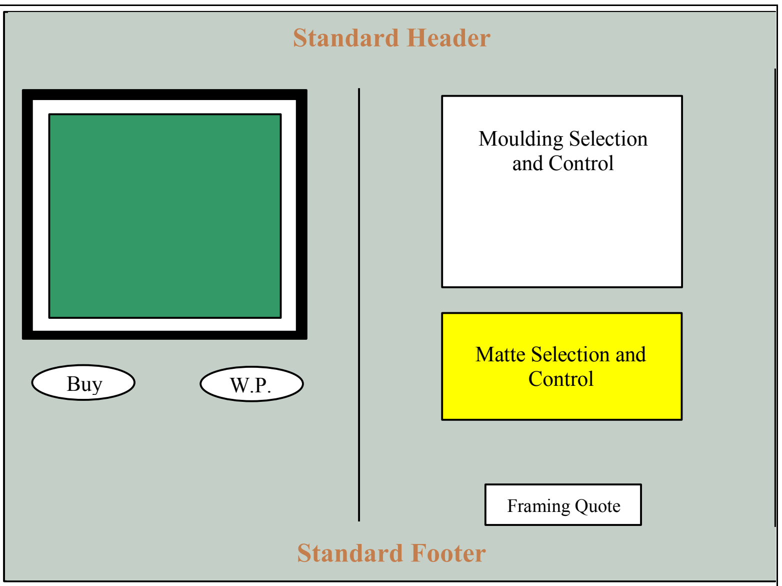
**Postcondition:** The framing and associated pricing information for the artwork are now displayed on the shopping cart page.

## User Interface Elements

Functional UI Elements provide details on key screen and user interface components. These items describe each element and it's functionality and provides an illustrative representation of the component. These drawings are only intended to convey the functionality needed. The look and feel as well as the method of implementation (dialog box, full screen, etc.) will be determined as part of a separate Look and Feel and usage model design effort.

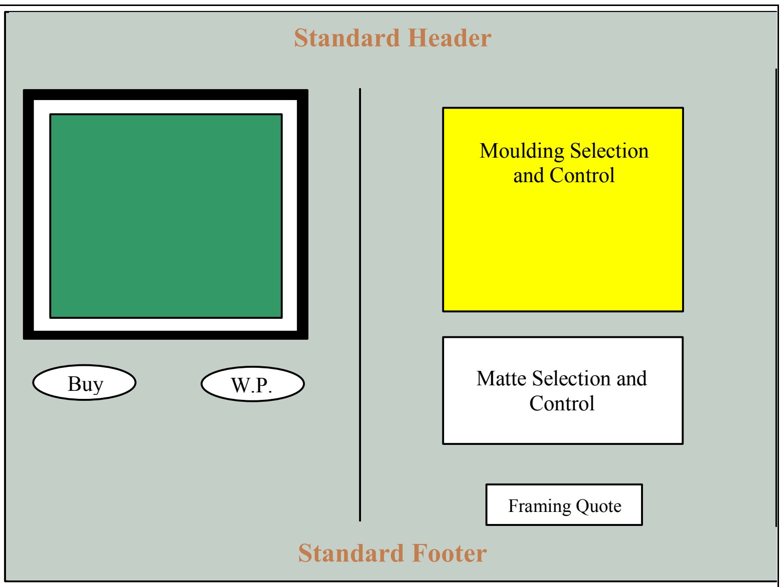
### 2.1 Mat Type/Color Selection Control

| | |
|---|---|
| Description | Matte control(s) |
| Functionality | Allows user to select or change the type and color of matting behind an artwork |
| Applies To | Framing page |
| Key Screen Elements | Matte selection and control, Reset button |
| General Appearance (illustrative) | |

| | |
|---|---|
| |   Standard Header<br>Moulding Selection and Control<br>Matte Selection and Control<br>Buy<br>W.P.<br>Framing Quote<br>Standard Footer |
| Other Comments | |
| Affected Pages | |
| Implementation Priority | |
| Implementation Difficulty | |
| Implementation Notes | |

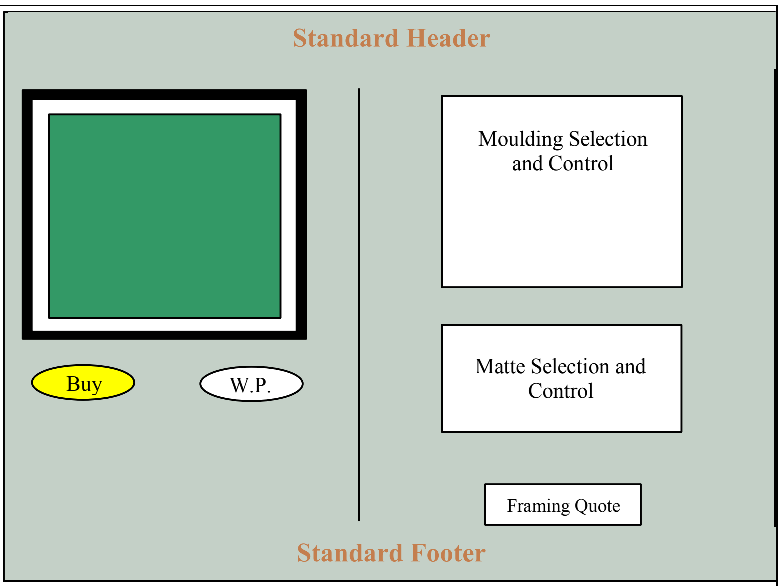
### 2.2 Moulding Selection Control

| | |
|---|---|
| Description | Moulding control(s) |
| Functionality | Allows user to select or change the moulding type and color for an artwork |
| Applies To | Framing page |
| Key Screen Elements | Moulding selection and control, Reset button |
| General Appearance (illustrative) | |

| | |
|---|---|
| |  Standard Header<br>Moulding Selection and Control<br>Buy<br>W.P.<br>Matte Selection and Control<br>Framing Quote<br>Standard Footer |
| Other Comments | |
| Affected Pages | |
| Implementation Priority | |
| Implementation Difficulty | |
| Implementation Notes | |

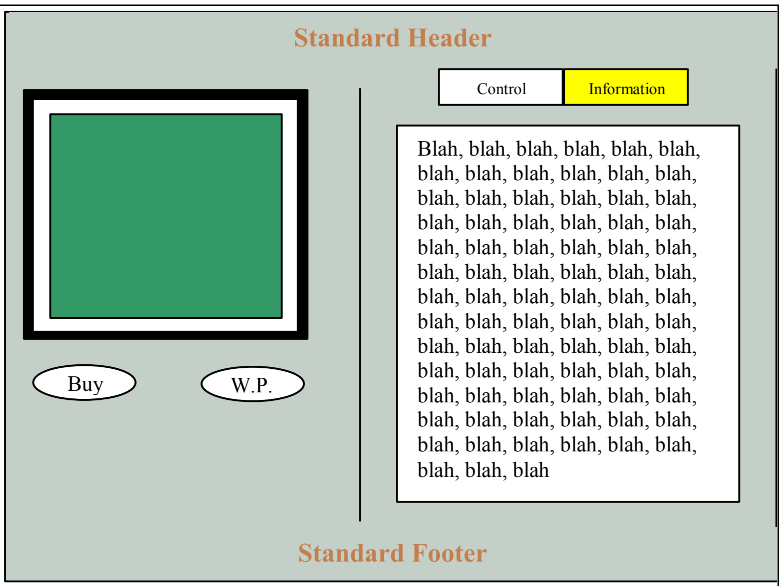
### 2.3 Purchase Art Button

| | |
|---|---|
| Description | Buy button |
| Functionality | Allows user to formalize the framing choices in an order |
| Applies To | Framing page |
| Key Screen Elements | Buy button |
| General Appearance (illustrative) | |

| | |
|---|---|
| |  Standard Header<br>Moulding Selection and Control<br>Buy<br>W.P.<br>Matte Selection and Control<br>Framing Quote<br>Standard Footer |
| Other Comments | |
| Affected Pages | |
| Implementation Priority | |
| Implementation Difficulty | |
| Implementation Notes | |

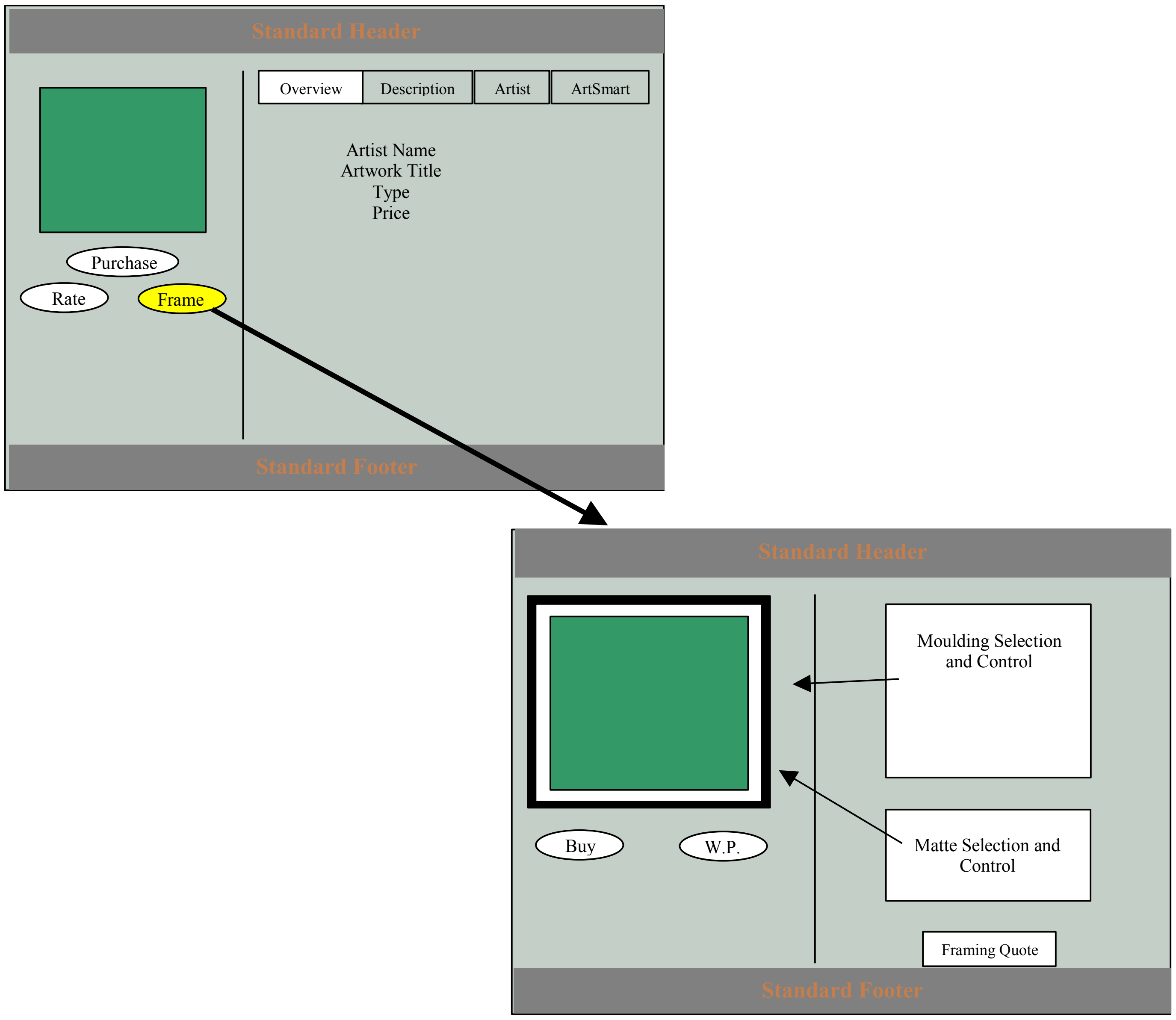
## 2.4 Information Tab

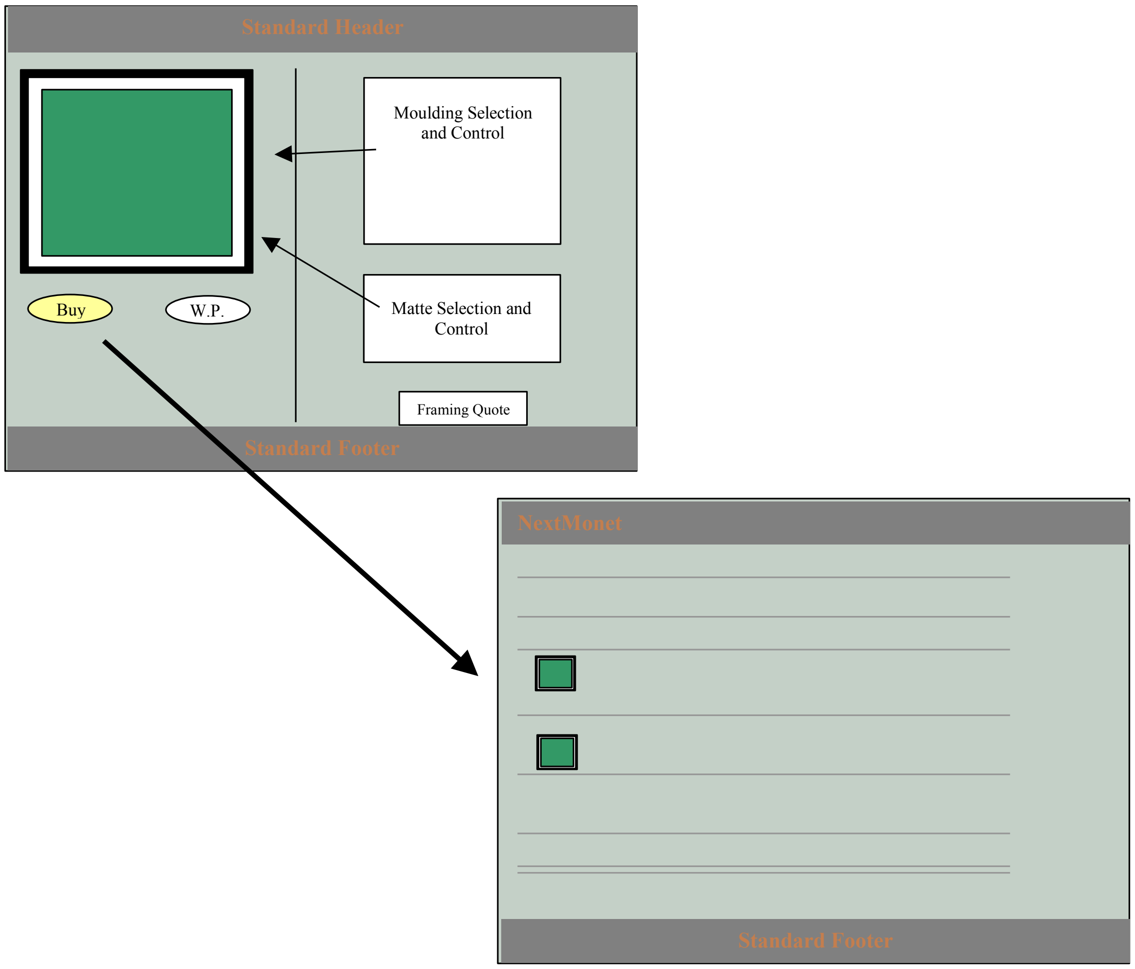
| | |
|---|---|
| Description | Buy button |
| Functionality | Allows user to formalize the framing choices in an order |
| Applies To | Framing page |
| Key Screen Elements | Buy button |
| General Appearance (illustrative) | |

| | |
|---|---|
| |   Standard Header<br>Control<br>Information<br>Buy<br>W.P.<br>Standard Footer Blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah, blah |
| Other Comments | |
| Affected Pages | |
| Implementation Priority | |
| Implementation Difficulty | |
| Implementation Notes | |

## Page Flows (Storyboards)

Page flows show how the user interacts with the framing model in a graphical way, and so show how the template development should progress.

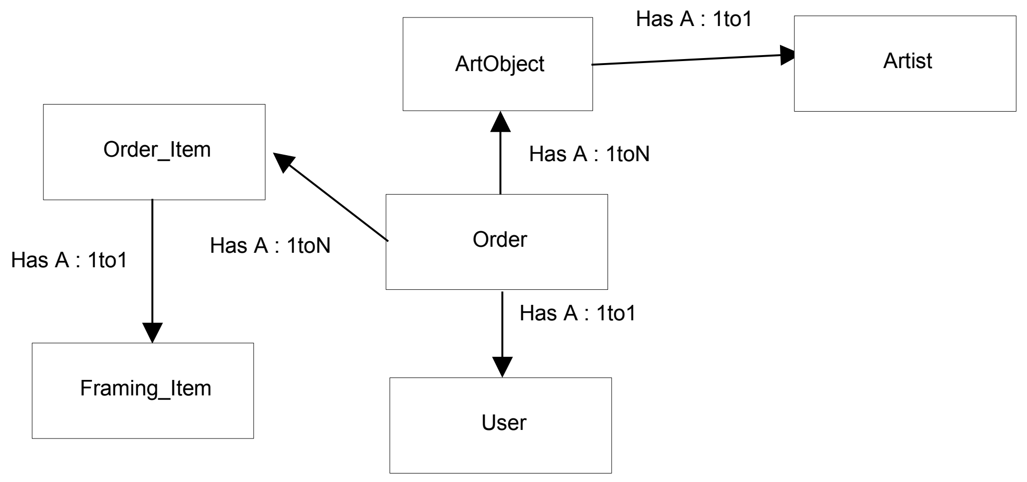

Standard Header
Overview
Description
Artist
ArtSmart
Artist Name
Artwork Title
Type
Price
Purchase
Rate
Frame
Standard Footer
Standard Header
Moulding Selection and Control
Matte Selection and Control
Buy
W.P.
Framing Quote
Standard Footer



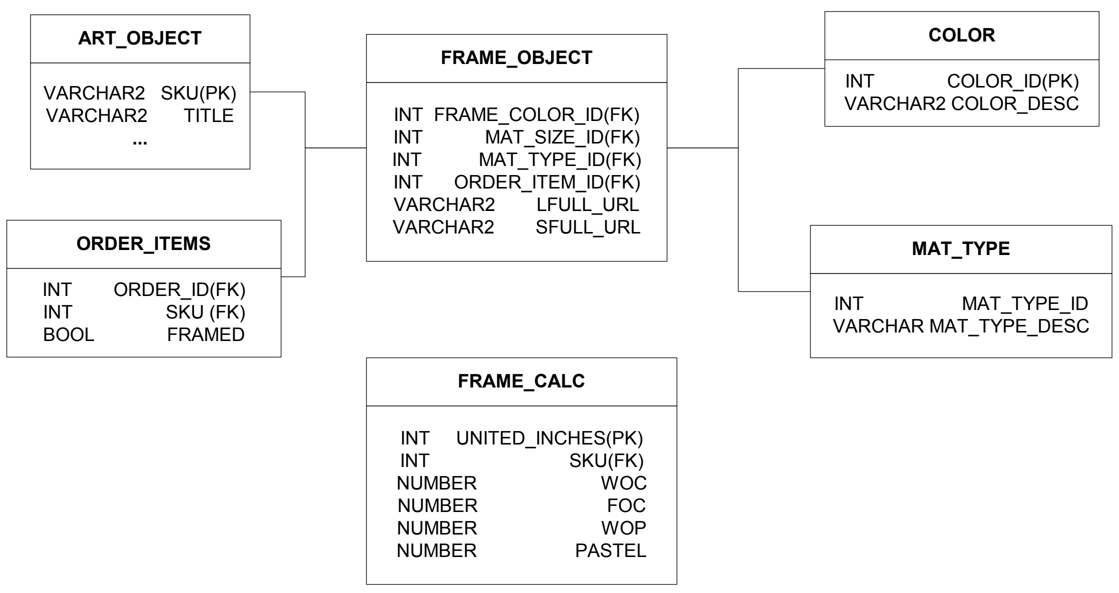
## The Object Models

The object models should show a graphical representation of the classes defined in each tool in an "object diagram" and the associated properties.  It should also start to talk about the data elements and properties of each object.

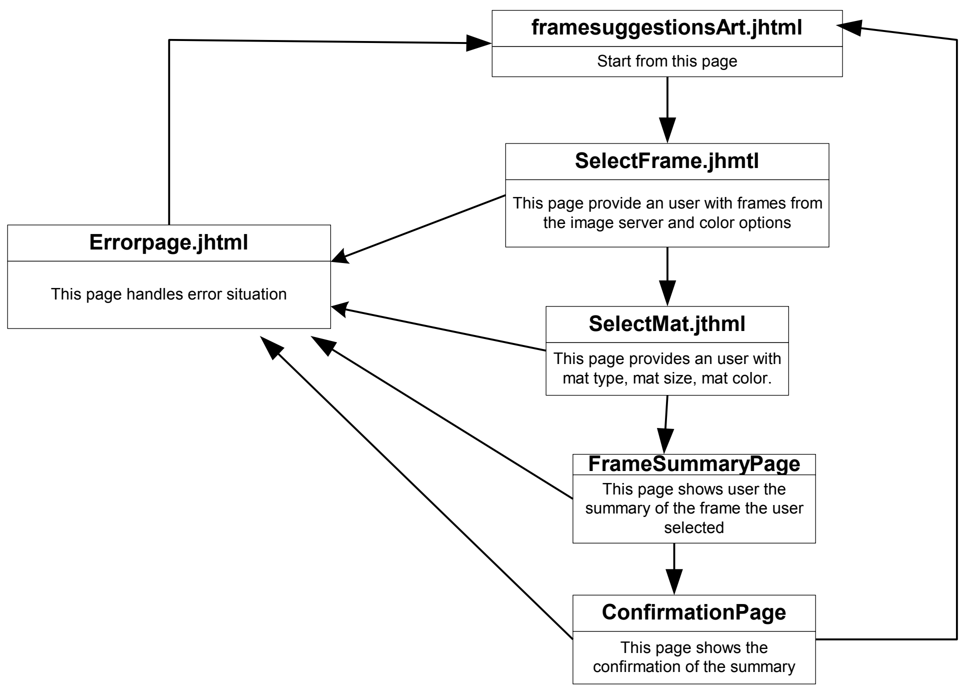
## Object Diagram – Framing Tool



### Schema Modifications



### Content Page Flow


framesuggestionsArt.jhtml
Start from this page
SelectFrame.jhmtl
This page provide an user with frames from the image server and color options
Errorpage.jhtml
This page handles error situation
SelectMat.jthml
This page provides an user with mat type, mat size, mat color.
FrameSummaryPage
This page shows user the summary of the frame the user selected
ConfirmationPage
This page shows the confirmation of the summary

## Implementation

The implementation of this project is comprised of three separate stages:

1. The TrueSpectra image server model
2. The data model to support framing quotes
3. The JTHML interface

### TrueSpectra Image Server Model

The development of the framing project is taking place using the TrueSpectra image server hosted on manet. The following components are being used:

- Layering model (.xml): "/data/Accelerate/images/product/frame/Frame.xml"
- Frame images (.jpg): "/data/Accelerate/images/product/frame/moldings/*.jpg"
- Mat images (.jpg): "/data/Accelerate/images/product/frame/moldings/whitesquare.jpg"
- Art images (.fps, .jpeg): "/data/Accelerate/images/product/images/*"

The model is invoked at the browser location textfield, or within an HREF tag, as follows:

http://localhost:8840/framing/frameArt.jhtml?sku=jco02002&pixelHeight=112&pixelWidth=84

These only get you to the starting page. The following is the URL that is sent to the image server. Note that it has a very long GET string. The engineers at TrueSpectra say that they can handle at least 1000-byte GET strings, but no one there has verified whether they can handle 2000+-byte GET strings.

http://manet:10666/product/frame/frame5.xml?cmp-image=picture,product/images/cbl01008.fpx&cmp-pos=picture,50.0,50.0,30.0,70.0&cmp-image=topleft,product/frame/moldings/topleft.jpg&cmp-pos=topleft,22.5,97.5,5.0,5.0&cmp-image=top,product/frame/moldings/top.jpg,keepheight&cmp-pos=top,50.0,97.5,50.3&cmp-image=topright,product/frame/moldings/topright.jpg&cmp-pos=topright,77.5,97.5,5.0,5.0&cmp-image=right,product/frame/moldings/right.jpg,keepwidth&cmp-pos=right,77.5,50.0,,90.2&cmp-image=bottomright,product/frame/moldings/bottomright.jpg&cmp-pos=bottomright,77.5,2.5,5.0,5.0&cmp-image=bottom,product/frame/moldings/bottom.jpg,keepheight&cmp-pos=bottom,50.0,2.52.5,50.3&cmp-image=bottomleft,product/frame/moldings/bottomleft.jpg&cmp-pos=bottomleft,22.5,2.5,5.0,5.0&cmp-image=left,product/frame/moldings/left.jpg,keepwidth&cmp-pos=left,22.5,50.0,,90.2&cmp-image=whitesquare,product/frame/moldings/whitesquare.jpg&cmp-pos=whitesquare,,,55.3,0.0&cmp-color=bkgdcolor,xeff4e4&cmp-end=1&cvt=jpeg

## Data Model

The data model for framing consists of a new FRAMING_CALC table, an associated FrameCalc Bean, a FrameCalcObjectDBManager, and a frame relational view and subviews.

## Logic Model

The framing logic is embedded in the Frame directory and consists of a FrameBean, FrameArtSB, FrameConstants, FrameGenericServlet, and FramingCalculation. The FramingCalculation object is responsible for calculating a framing quote and makes use of the Art_Object and Framing_Calc tables and associated beans. The independent variable for framing is the Art_Object sku. This is used to retrieve the object, whereupon the type of art object is parsed from the Short_Description field as one of: (1) work on canvas, (2) floater on canvas, (3) work on paper, or (4) pastel. Likewise, the physical dimensions of the object are obtained to arrive at the frame size. These two pieces of information are used to calculate the framing cost. The logic for the framing display is also embedded in the Frame directory, in the FrameArtSB object.

## JHTML Interface

The primary interface for the framing project is the new webpage, frameArt.jhtml. The lefthand side of the window is displayed using artImage.jhtml. The righthand side of the window is build from the framing controls, selectOptions.jhtml, and various zooming controls for the frame sections (zoomFrameType.jhtml), and from an information panel, framingInfo.jhtml.