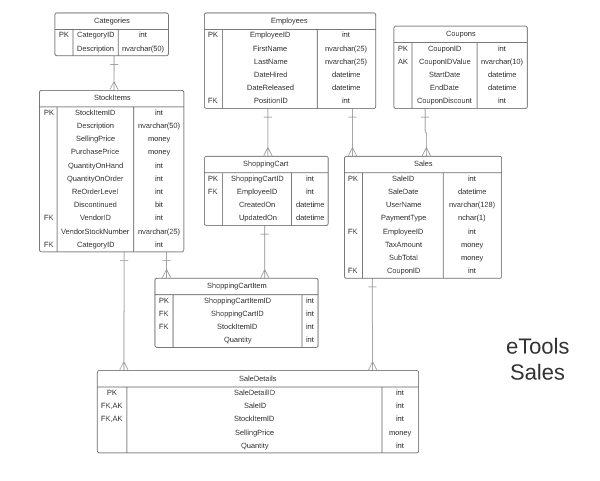
Sales

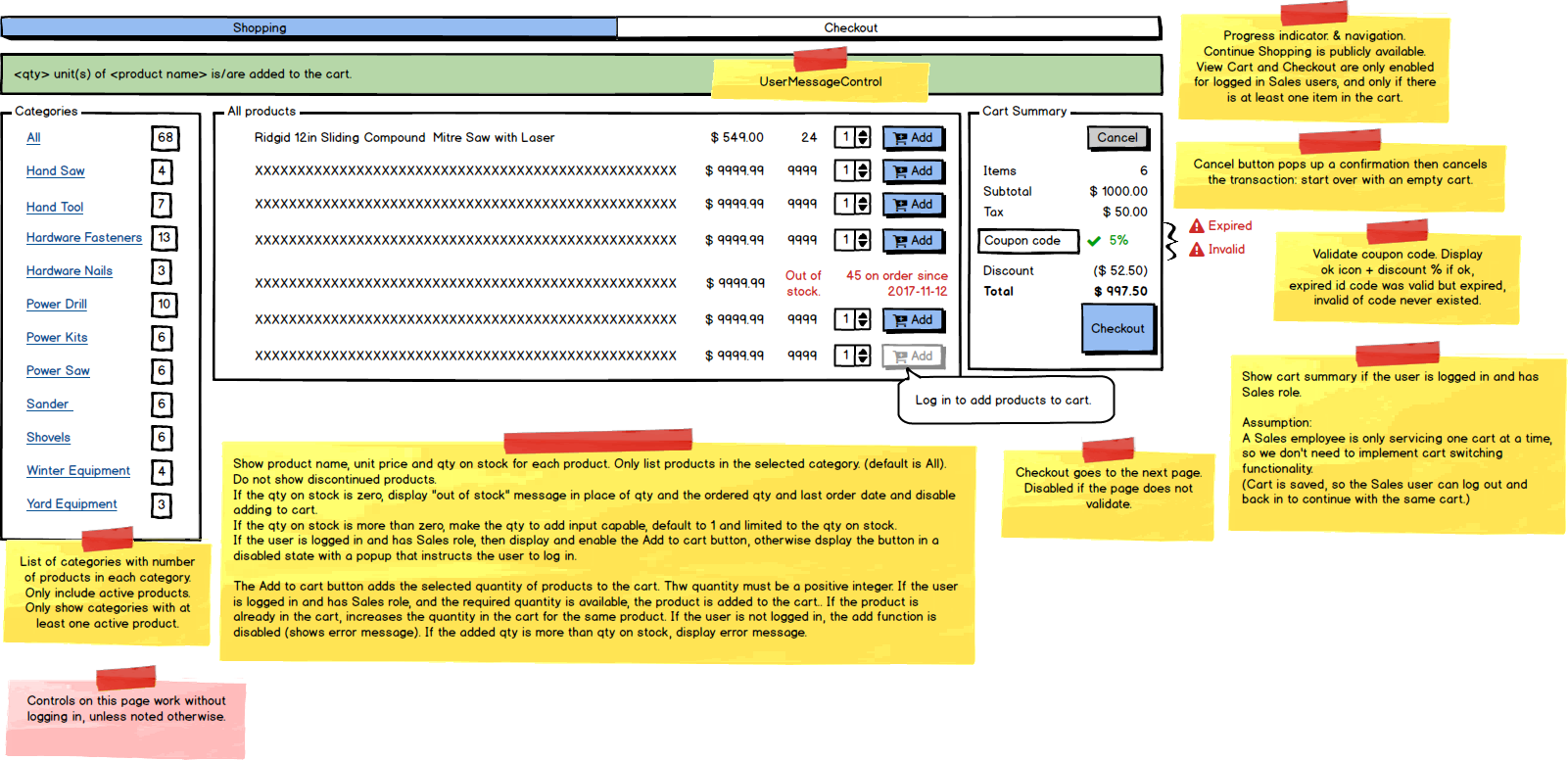
# ERD for UX process



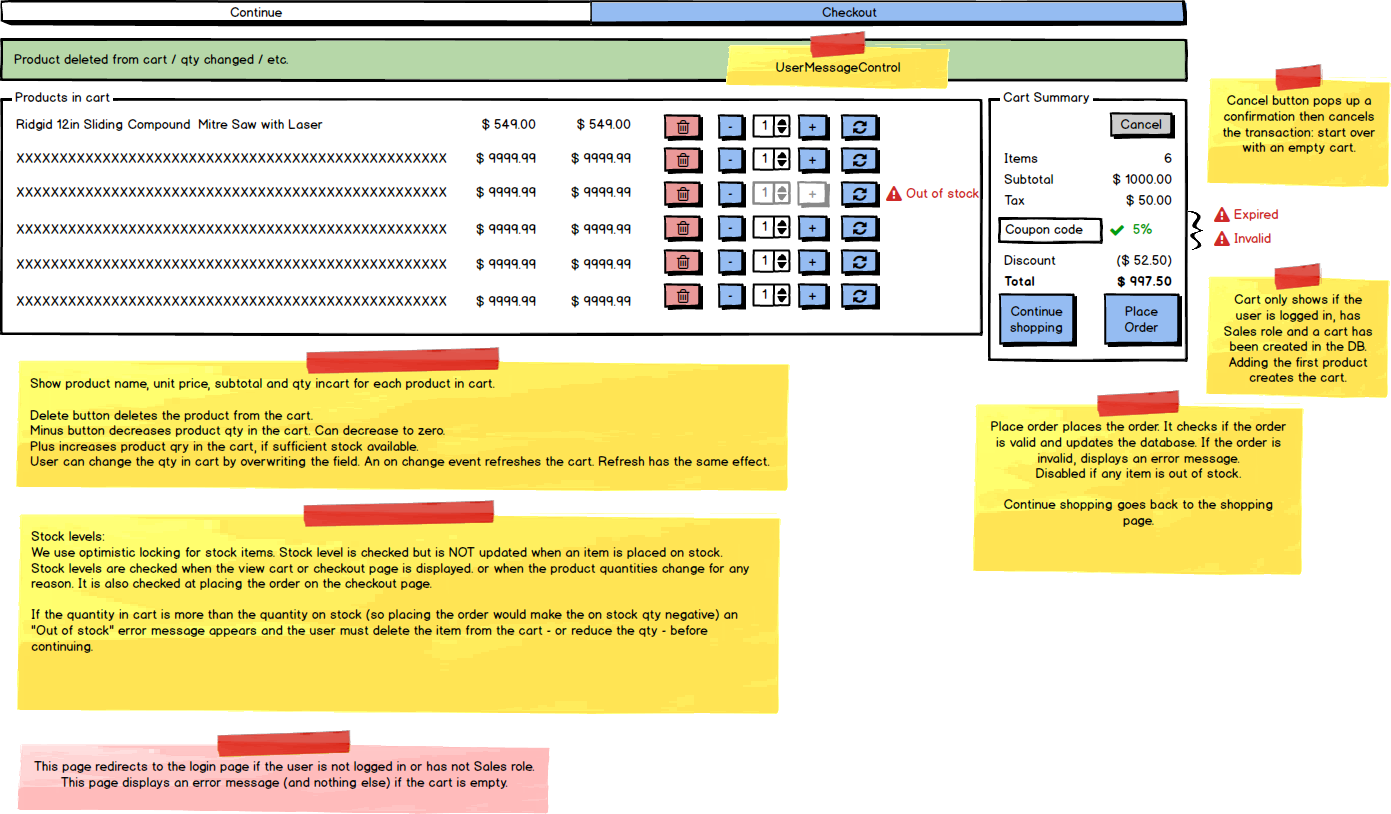
# Possible UX Interface

I’m planning to implement the Sales subsystem on a single page using tabbed pages (asp:MultiView).

## Shopping view (initial view)



## Checkout view



# Databound Controls

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| View | Control | Type | Entity | Columns |
| Shopping | CategoryList | ListView | CategoryInfo (P) | CategoryID (hidden)  CategoryDescription  ProductsInCategory |
| Shopping | ProductList | ListView | ProductInfo (P) | ProductID (hidden)  ProductDescription  UnitPrice  QtyOnHand  QtyOnOrder  PendingOrderDate  IsLowStock |
| Shopping Cart  Checkout | CartSummary | FormView | CartSummaryInfo (P) | CartID (hidden)  ProductsInCart  Subtotal  Tax  CouponInfo  Discount  Total  isCartValid |
| Shopping  Cart  Checkout | Coupon | TextBox + Labels | CouponInfo(P) | CouponID (hidden)  CouponCode  Discount  isValid  ErrorMessage |
| Cart  Checkout | ProductsInCart | ListView | ProductInCartInfo(P) | ProductID (hidden)  ProductName  UnitPrice  Qty  Total  isOutOfStock  IsLowStock |

## State Persistence

|  |  |
| --- | --- |
| **State Information** | **Persisted in** |
| Current tab | Only in view state (MultiView selectedIndex) |
| Selected product category | Only in view state (ListView selectedIndex) |
| Shopping cart | ShoppingCart database table |
| Products and quantities in the shopping cart | ShoppingCartItem database table |
| Logged in user ID and user role | User entity (managed by AppSecurity subsystem) |
| Coupon code | Only in view state (in Coupon control) |

# BLL Designs

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| CategoryList | | None / Wired via ODS |
| **BLL Class(es) and Method (s)** | | |
| SalesCategoryController | List<CategoryInfo> FindAllCategories()  *Retrieve a list of categories along with number of distinct products in the category. Only return categories with at least one product.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Read: Categories, StockItems  from p in StockItems  where !p.Discontinued  group p by p.Category into grp  select new CategoryInfo  {  CategoryID = grp.Key.CategoryID,  CategoryDescription = grp.Key.Description,  ProductsInCategory = grp.Count()  }; | | CategoryInfo(P)  public class CategoryInfo  {  public int CategoryID;  public String CategoryDescription;  public int ProductsInCategory;  } |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| ProductList | | None / Wired via ODS |
| **BLL Class(es) and Method (s)** | | |
| SalesProductController | List<ProductInfo> FindProductsByCategory(Int CategoryID)  *List all active (not discontinued) products in the given category. If CategoryID is null or zero, retrieve all products. Returns empty result set if CategoryID is invalid or no active products in the category.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Read:  Categories, StockItems, ShoppingCartItems, PurchaseOrderDetails, PurchaseOrders  from p in StockItems  where !p.Discontinued  select new ProductInfo  {  ProductID = p.StockItemID,  ProductDescription = p.Description,  SellingPrice = p.SellingPrice,  QtyOnHand = p.QuantityOnHand,  QtyOnOrder = p.QuantityOnOrder,  QtyInCart = (from ci in p.ShoppingCartItems select ci.Quantity).Sum(),  PendingOrderDate = (from po in PurchaseOrderDetails where !po.PurchaseOrder.Closed select po.PurchaseOrder.OrderDate).Min()  }; | | ProductInfo(P)  public class ProductInfo  {  public int ProductID {get; set;}  public String ProductDescription {get; set;}  public Decimal SellingPrice {get; set;}  public int QtyOnHand {get; set;}  public int QtyOnOrder {get; set;}  public int? QtyInCart {get; set;}  public DateTime? PendingOrderDate {get; set;}  public bool isLowStock {  get { return (QtyInCart != null || QtyInCart >= QtyOnHand); }  }  } |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| AddProductToCartButton | | AddProductToCart  *Retrieve username, product ID, qty, then call SalesCartController.AddToCart*  *Navigate to the shopping tab.*  *Reset the product filter to All.* |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | void AddToCart(string username, int productId, int quantity)  *Adds the product with the given quantity into the cart under the given user and decreases the stock on hand for the given product.*  *If the product ID is invalid, or the product has not enough stock on hand, it raises an exception. Also raises an exception if the user has no Sales Role.*  *Actions:*   * *If the user has no cart yet, it creates the cart first, otherwise updates the last update timestamp of the cart* * *If the product already exists in the cart, increases the qty in the cart.* * *If the product is not yet in the cart, inserts the product into the cart.* * *Decreases the stock on hand for the product.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Read:  StockItems  Update:  StockItems  Create or Update:  ShoppingCartItems, ShoppingCarts | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| CartSummary | | None / Wired via ODS |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | CartSummaryInfo GetCartSummary(String username, String couponCode)  *Retrieve shopping cart for the given username and calculate discounts based on the couponCode.*  *Returns null if the user has no shopping cart or the username is not found.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Read:  ShoppingCartItems, ShoppingCarts,  Coupons | | CartSummaryInfo |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| CancelButton | | CancelCart *Clear the coupon code control.*  *Navigate to the shopping tab.*  *Reset the product filter to All.* |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | void CancelCart (String username)  *If a shopping cart exists for the user, then delete the shopping cart items and the shopping cart.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Delete (if exists):  ShoppingCartItems, ShoppingCarts | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| CheckoutButton  CheckoutTabButton | | Checkout  *Navigate to the checkout tab.*  *Display error if user not authorized, cart not exists or user not authorized* |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | boolean checkCartExists(string username)  *Checks if the cart exists and it has at least one product in it.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Delete (if exists):  ShoppingCartItems, ShoppingCarts | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| ProductsInCart | | None / Wired via ODS |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | List<ProductInCartInfo> FindProductsInCart(String username)  *Returns products in the current user’s cart or empty list if the user has no cart or if it has no products.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Read:  ShoppingCartItems, ShoppingCarts,  StockItems | | ProductInCartInfo |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| DeleteProductFromCart | | DeleteProductFromCart |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | void DeleteProduct(String username, int productID)  *Deletes the selected product from the cart.*  *Exception if the cart has no such product, no cart or user not authorized.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Read:  ShoppingCartItems, ShoppingCarts | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| IncreaseProduct | | IncreaseProduct  *Increases QTY then calls Refresh* |
| **BLL Class(es) and Method (s)** | | |
| n/a | n/a | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| n/a | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| DecreaseProduct | | DecreaseProduct *Decreases QTY then calls Refresh* |
| **BLL Class(es) and Method (s)** | | |
| n/a | n/a | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| n/a | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| UpdateCartItem | | UpdateCartItem |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | void UpdateCartItem(string username, int productId, int newQuantity)  *Updates the product to the new quantity in the cart.*  *Exception if not enough stock, product not in cart, cart not exists or user not authorized.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Read:  StockItems, ShippingCart  Update:  ShoppingCartItems | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| PlaceOrderButton | | PlaceOrder |
| **BLL Class(es) and Method (s)** | | |
| SalesCartController | void PlaceOrder(string username)  *Checks if the user has Sales role, has cart and cart has items.*  *Checks that each cart item has sufficient stock.*  *(If any check fail, throws appropriate exception)*  *In a transaction:*   * *Decreases qty on hand in StockItems* * *Records Sales and SaleDetails* * *Deletes the ShoppingCart and ShoppingCartItems*   *Rolls back the transaction and raises exception if error occurs.* | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| Create:  Sales, SaleDetails  Delete:  ShoppingCart, ShoppingCartItem  Update:  StockItems | | n/a |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| ShoppingButton  ShoppingTabButton | | Shopping  *Navigate to the shopping tab.* |
| **BLL Class(es) and Method (s)** | | |
| n/a | n/a | |
| **SQL Table(s): (C,R,U,D)** | | **Entities/DTOs/POCOs** |
| n/a | | n/a |

Comment

Excellent work as I have come to expect from you.