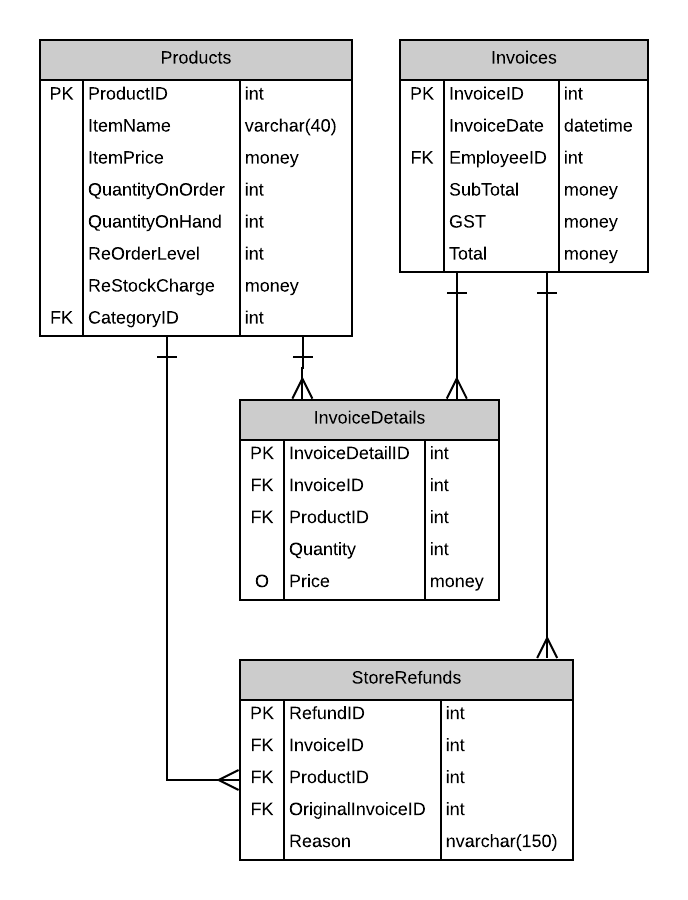
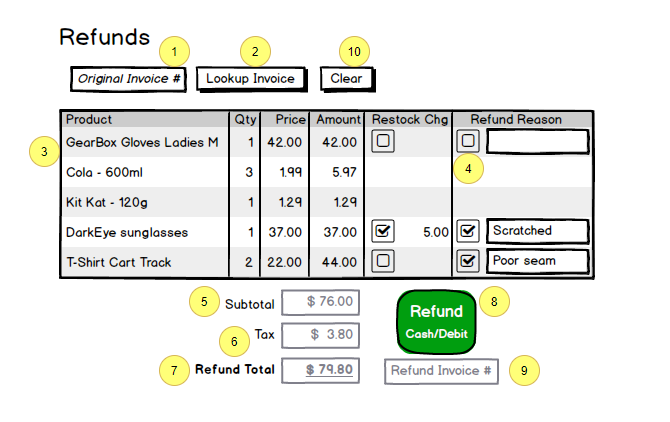
Sales Subsytem (Subject to change)

# ERD for UX process



# Possible UX Interface





# **Purchasing Page**

# Event and Wiring Summations

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| EmployeDDL(1) (EmployeeID)  CategoryProductDDL(2) (CategoryID)  ProductDDL(3) (ProductID) | | None / wired via ObjectDataSource   * Dropdown will only be populated after an Employee with **CLERK** role logs in to the system * Select input for CategoryProductDDL * Loads ProductDDL with products from that category when a list item is selected in CategoryProductDDL |
| **BBL Class(es) and Method (s)** | | |
| EmployeeController  CategoryController  ProductController | List\_ClerkEmployees(int categoryid)   * List qualified employees to work on the sales purchasing   List\_Categories()   * Lists all categories   List\_GetProductByCategory(int categoryid)   * Loads products from the selected Category | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
| Categories (R), Products (R) | | SelectionList |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| QuantityArg(4) (TextBox)  AddToCart(5) (LinkButton)  SubtotalTextBox(8) (TextBox)  TaxTextBox (9) (TextBox)  TotalTextBox (10) (TextBox) | | OnClick   * QuantityArg will be the input of how much of that product is wanted and validate if there are no quantities * Re-adding will increase quantity of product * Values will be added to Subtotal, Tax and Total |
| **BBL Class(es) and Method (s)** | | |
| ALL DONE IN CODEBEHIND | ALL DONE IN CODEBEHIND | |
| **Code Behind** | | |
| **Add Product(s) into Gridview**   * Use ShoppingCartList to hold information about products * Dump contents of ShoppingCartList using a foreach loop to access gridview rows * Add instance to ShoppingCartList * DataSource and DataBind the list to the gridview   **Calculations**   * Item.Quantity \* Item.Price = Item.Amount * Subtotal += item.Amount * Tax = Subtotal \* 0.05M * OrderTotal = Subtotal + tax | | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
| Categories (R), Products (R) | | ShoppingCartList (Create instances for gridview) |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| CartGridView(6) (ItemCommand)  SubtotalTextBox(8) (TextBox)  TaxTextBox (9) (TextBox)  TotalTextBox (10) (TextBox) | | OnItemCommand   * Delete a product in the ShoppingCartList by ProductID * Values of subtotal, tax and total will be updated to the current price |
| **BBL Class(es) and Method (s)** | | |
| ALL DONE IN CODEBEHIND | ALL DONE IN CODEBEHIND | |
| **Code Behind** | | |
| **Remove Product(s) from Gridview**   * Access contents of gridview rows using foreach loop * Find ProductID that needs to be removed * DataSource and DataBind the list current ShoppingCartList to the gridview for update   **Calculations**   * Item.Quantity \* Item.Price = Item.Amount * Subtotal += item.Amount * Tax = Subtotal \* 0.05M * OrderTotal = Subtotal + tax | | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
|  | | ShoppingCartList |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| RefreshButton(7) (LinkButton)  SubtotalTextBox(8) (TextBox)  TaxTextBox (9) (TextBox)  TotalTextBox (10) (TextBox) | | OnClick   * Refresh will increase the quantity of the product * Values of subtotal, tax and total will be updated to the current price |
| **BBL Class(es) and Method (s)** | | |
| ALL DONE IN CODEBEHIND | ALL DONE IN CODEBEHIND | |
| **Code Behind** | | |
| **Add Product(s) into Gridview**   * Use ShoppingCartList to hold information about products * Dump contents of ShoppingCartList using a foreach loop to access gridview rows * Add instance to ShoppingCartList for updated quantities * DataSource and DataBind the list to the gridview   **Calculations**   * Item.Quantity \* Item.Price = Item.Amount * Subtotal += item.Amount * Tax = Subtotal \* 0.05M * OrderTotal = Subtotal + tax | | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
|  | | ShoppingCartList |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| PaymentButton (12) (LinkButton)  InvoiceNumberTextBox (11) (TextBox) | | OnClick   * Process gridview contents and create an invoice * Update quantity on hand for products in the inventory * Invoice remain in screen when complete, and display invoice number on screen (11) |
| **BBL Class(es) and Method (s)** | | |
| InvoiceController | Public int CreateInvoice(decimal gst, decimal gst, decimal order, decimal ordertotal, List < ShoppingCartList > cartlist)  **TRX**   * Create variable of Invoice entity which queries new invoice created * If exists is null   + Create new instance of Invoice * Else if is Error.Count > 0   + Validate cannot add an existing invoice   + Throw error | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
| Invoices(C), Products(U), InvoiceDetails(C) | |  |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| Clear (13) (LinkButton) | | OnClick   * Clear / Reset Form * Gridview is not displayed when clicked |
| **BBL Class(es) and Method (s)** | | |
|  |  | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
|  | |  |

# **Refund Page**

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| OriginalInvoice (1) (TextBox)  LookUpInvoice (2) (LinkButton) | | OnClick   * Lookup invoice that has been created |
| **BBL Class(es) and Method (s)** | | |
| InvoiceDetails | Public <InvoiceDetails> GetInvoiceDetails(int invoiceid)   * Query that finds generated Invoice   Binds to InvoiceListGridView | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
| InvoiceDetails(R) | |  |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| InvoiceListGridView (3) (GridView)  RefundReasonCheckBox (4) (CheckBox)  RefundReasonTextBox (5) (TextBox)  SubtotalTextBox (6) (TextBox)  TaxTextBox (7) (TextBox)  RefundTotalTextBox (8) (TextBox) | | CheckChanged   * Select which product to be returned * Validate the reason why it needs to be returned (5) * Populate the values of subtotal, tax, and refundtotal (6,7,8) |
| **BBL Class(es) and Method (s)** | | |
|  |  | |
| **Code Behind** | | |
| **Select Product to be removed**   * Access the gridview via foreach loop * Find the item that is checked and validate the reason if empty   **Calculations**   * Subtotal += item.amount – restockcharge * Tax = Subtotal \* 0.05M * RefundTotal = Subtotal + tax | | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
|  | |  |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| RefundReasonCheckBox (4) (CheckBox)  RefundButton (9) (LinkButton) | | OnClick   * Return products back to the store |
| **BBL Class(es) and Method (s)** | | |
| InvoiceController | ReturnProducts(int originalinvoice, decimal subtotal, decimal gst, decimal total, List<ReturnedProducts> returnedproducts)  **TRX**   * Create variable of Invoice entity which queries new invoice created * Add current invoice to a new invoice * If exists is null   + Create new instance of StoreRefund   + Select each item that needs to be returned * Else if is Error.Count > 0   + Validate cannot add an existing invoice * Throw error | |
| **Code Behind** | | |
| **Refund Clicked**   * Foreach loop gridview to see which items that needs to be returned via RefundReasonCheckBox(4) * Create instance of ReturnedProducts and see which items needs to be returned * And add into ReturnedProducts | | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
| Invoices(C,U), Products(U), InvoiceDetails(C,U) | | ReturnedProducts |

|  |  |  |
| --- | --- | --- |
| **Controls** | | **Events** |
| ClearButton (9) | | OnClick   * Clears webform * Gridview is not displayed |
| **BBL Class(es) and Method (s)** | | |
|  |  | |
| **Code Behind** | | |
|  | | |
| **SQL Table(s): (C,R,U,D)** | | **ViewModels** |
|  | |  |

# **View Models**

**SelectionList**

{

Int ValueID {get;set;}

string Displaytext {get;set;}

}

**ShoppingCartList**

{

int ProductID { get; set; }

string ProductName { get; set; }

int Quantity { get; set; }

decimal Price { get; set; }

[NotMapped]

public decimal Amount

{

get

{

return Price \* Quantity;

}

}

}

**ReturnedProducts**

{

Int InvoiceID {get;set;}

Int productID {get;set;}

Int quantity {get;set;}

String reason {get;set;}

}