Project 2.1
Problem Analysis

Overview:
While large scale retail stores have the resources to create a commanding web presence, smaller companies do not have the desire or budget to create a unique website. In order for these smaller companies to be able to compete in the web market, they require existing solutions that can be chained together into a functioning website. This project aims to create a shopping card implementation that can be plugged into any client website with minimal effort.

The shopping cart will mimic the user's model of an in-store shopping cart: items can be added, removed, changed, and at the end the user can check-out. In addition, the cart should be tied to the user so that moving from page to page does not affect the contents of the cart. This implementation will appear to be very similar to the Amazon.com or Newegg.com shopping cart, but will be an open solution that can be used as a plugin for an existing site.

Because the cart will be handling user payment information, security and consistency are critical. If the end-user ever believes that his payment information may be mishandled, he will not return to the website and the small business will lose customers. Thus the cart should use strong encryption and rely on user authentication to protect credentials. The cart must also implement an "exactly once" model for payment submission. This means that if the user refreshes a page, the order will not be processed twice or zero times; the payment and order request must occur exactly one time.

Object Model, Event Model, and Wireframes are attached separately.

Behavior:
The user will be able to add and change purchase requests in his cart by perusing the business's inventory and choosing items to purchase. Adding items to the cart will indicate to the user that the item has been added in a given quantity, but will not take the user to the shopping cart page. The user can view the shopping cart summary by clicking a link to the cart, and from there is able to modify and remove items and eventually checkout. Checkout redirects the user to a separate payment site and indicates to him that the order will be processed when the payment is received. In this iteration of the product, Checkout causes the shopping cart to be blown away and ***does not*** change the inventory of the shopkeeper. This is intentional. The inventory should not change until the payment has been processed and the inventory is leaving the warehouse. Since the merchant is not processing the payment in this model, it is wrong for the inventory to be automatically decreased.

For the administrator, there is a summary page that allows products to be added, removed, and changed with prices and inventory. The administrator will also be able to view the orders table.

Because users will not feel comfortable making purchases if they feel their information is insecure, the cart will enforce password authentication through unique email sign-ins. The cart will also protect businesses by preventing unauthenticated users from modifying prices, inventory, or viewing the orders that are currently being processed (this would constitute a privacy violation for the customers).

The admin user has default username and password “admin, admin”.