Jennifer Liu
6.170
11/6/12

Project 2 Problem Analysis:
J-Cart Shopping Cart

Overview

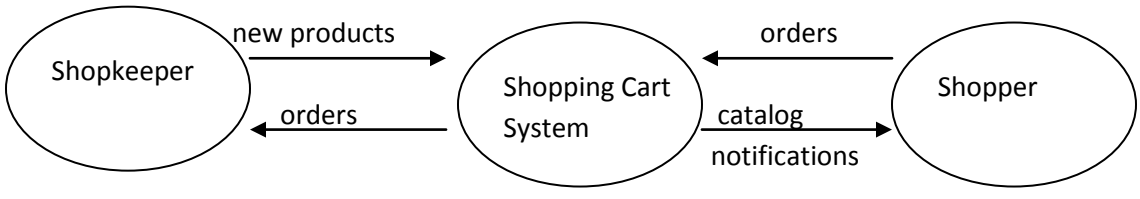
Purpose and goals

The goal of the system is to provide an easy way for online shoppers to aggregate the items they want to purchase in into a virtual shopping cart so that they can "checkout" and purchase multiple items all at once. The shopping cart will persist throughout a session, represented by an icon with the subtotal in the upper corner, and may even be saved for later sessions if the user wishes. The user may change items inside the shopping cart, editing quantities and deleting items before they proceed to checkout. The checkout should be as smooth and painless as possible, with the user selecting a previously entered address or entering a new one, reviewing their order, and then placing the order.

Meanwhile, the shopkeeper has an easy way to manage products and incoming order. The main page will display a summary of recent orders, with each order expandable to show specific item details. The shopkeeper will also have a smooth interface for managing products that are currently in stock, having the ability to add, modify, and delete products.

Many shopping carts exist on the web in various forms. This cart aims to simplify the ordering process as much as possible, making it not only easy for the shopper to purchase a bundle of items, but also easy for the shopkeeper to manage orders, keeping track of which ones have been shipped and which ones have been completed.

Context diagram



Unusual requirements

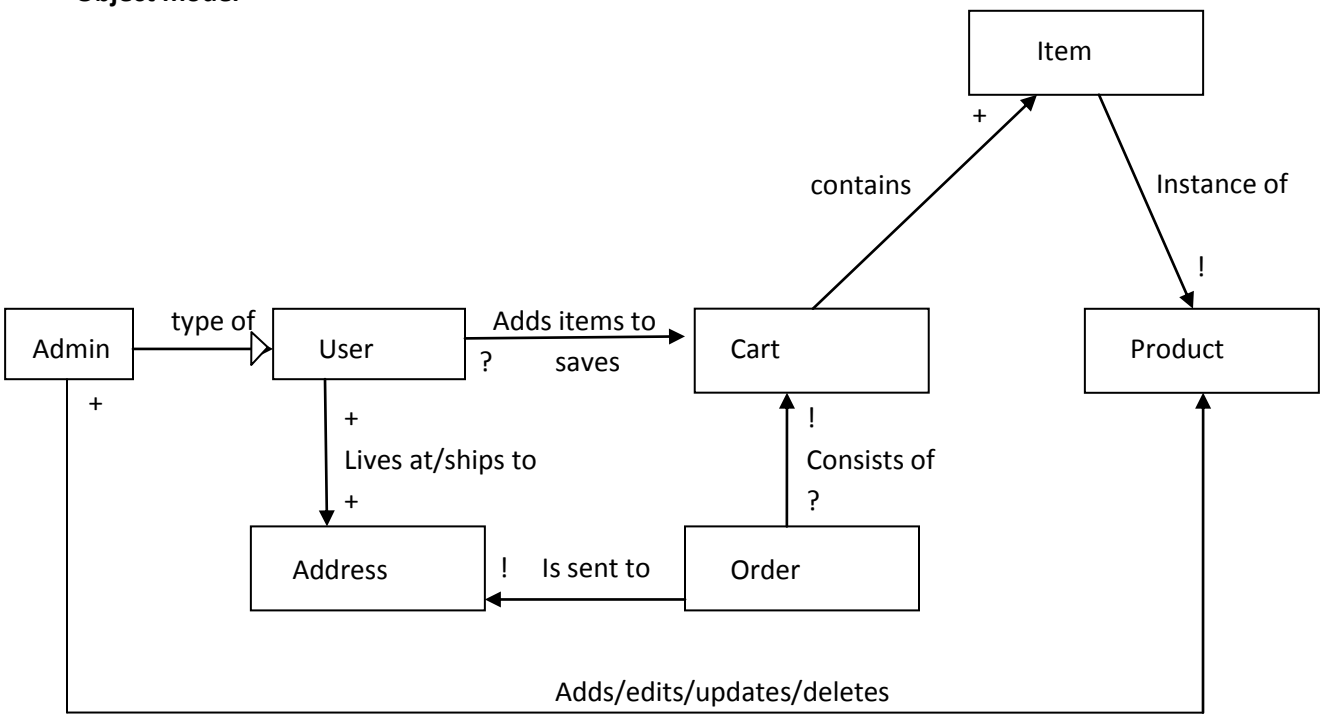
If purchases are to be incorporated into the system, a secure payment method is needed to prevent credit card theft or fraud. Some sort of encryption will be needed to protect credit card numbers, which could be accounted for by a third-party service such as PayPal.

Key features

- User interface for shopper including ability to add and remove items from cart, view/edit cart, select a shipping address, proceed to checkout and place order
- User interface for shopkeeper, including ability to create and modify catalog of items and prices, and to view recent orders from customers
- Ability to save shopping carts and re-activate them for later sessions and purchase

Domain

Object model



An Order corresponds (or "belongs to") to a Cart filled with Items that the user has decided to purchase. When an Order is created, the status of the corresponding Cart becomes "ordered".

Admin is a subclass of User, with all of the same attributes. The classification of type Admin, however, grants a User special privileges to ask the admin portion of the site, where Products may be added, edited, updated, and deleted.

Event model

System events:

Register: new user signs up for an account

Shop: user performs one or more of the following actions, in any order:

Add_to_cart: adds item to shopping cart

Edit_quantity: changes the quantity of item in shopping cart

Remove_from_cart: removes the item from the shopping cart entirely

Login: user logs into system

Save_cart: saves the current cart for later, creates a new (empty) cart

Checkout: user places order for everything in the cart

Logout: user logs out of system

Add_to_catalog: admin adds new product to catalog

Edit_product: admin edits product information

Remove_product: admin removes product from catalog

Environment events:

Restock ::= product quantity is increased

Possible event orderings:

User ::= [Shop] register (login (Shop (save_cart | checkout))* logout)*

Shop ::= (add_to_cart | edit_quantity | remove_from_cart)*

Product ::= add_to_catalog (add_to_cart edit_quantity* (remove_from_cart | checkout)) remove_product

Admin ::= (add_to_catalog edit_product* remove_product)*

Loosening of event order:

> The product may be edited by the shopkeeper while it is present in a user's cart, for example, if the product is restocked. In case of a browser crash, the user's event sequence may be truncated.

Behavior

Feature descriptions

- **Shopper interface**: allows the user to select items while browsing the site and view all selected items at once. User may view the cart and change quantities within the cart or remove items. User may then select a shipping address and then place an order.
- **Shopkeeper interface**: allows shopkeeper to see all recent orders, and mark which ones have been shipped. Will be sorted by most recent non-shipped order first. Shopkeeper may add new products, change their quantities, edit their information, and delete them.
- **Shopping carts**: Maybe be saved for checkout at a later date.

Security concerns

> Both shopper and shopkeeper authentication must be secure. Each admin page must verify that the user who is logged in is an admin, otherwise it should not display the page. Pages that contain information specific to the user must also check the identity of the user before displaying, otherwise hackers could easily manipulate URLs to access other users' information. Rails' has_secure_password feature, which encrypts passwords, should keep user login information safe, while filters may be used to verify authentication on each page.

Operations

- **Register** (POST /users)
  - Requires: valid user params, password matches confirmation
  - Modifies: users
  - Effects: new user created, user_id saved in cookies

- **Login** (POST /sessions)
  - Requires: valid email and password
  - Modifies: cookies
  - Effects: stores user id in cookies if authentication is successful

- **Logout** (DELETE /sessions)
  - Requires: user is logged in
  - Modifies: cookies
  - Effects: deletes user id and cart id from cookies

- **Add_to_cart** (GET /products/id/add_to_cart)
    - Requires: product with id exists
    - Modifies: item, product
    - Effects: if product is in stock, create a new cart if there isn't one currently active, and add 1 item of the product to the cart

- **Edit_quantity** (POST /items/edit)
    - Requires: valid item id and qty, where qty >0
    - Modifies: item, product
    - Effects: item and product quantities are changed if product.qty_in_stock > qty

- **Remove_from_cart** (GET /items/id/remove)
    - Requires: item with id exists
    - Modifies: item, product
    - Effects: product stock is restored with quantity of item, item is destroyed

- **View_current_cart** (GET /carts/current)
    - Requires: user is logged in

- **View_saved_carts** (GET /carts/saved)
    - Requires: user is logged in

- **Checkout** (GET /carts/id/checkout)
    - Requires: user is logged in, cart with id exists and belongs to user

- **Place_order** (POST /orders)
    - Requires: valid cart id
    - Modifies: cart, orders, cookies
    - Effects: Sets cart's status to "ordered", creates a new order associated with the cart, and removes cart from cookies

- **Save_cart** (GET /carts/id/save)
    - Requires: user is logged in, cart with id belongs to user, cart.status = "current", cart is not empty
    - Modifies: cart
    - Effects: Sets cart status to "saved"

- **Activate_cart** (GET /carts/id/activate)
    - Requires: user is logged in, cart with id belongs to user, cart.status = "saved"
    - Modifies: cart
    - Effects: Saves currently active cart if there is one, and activates cart with specified id by setting its status to "current"

- **Add_to_catalog** (POST /products)
    - Requires: Admin is logged in, valid product params
    - Modifies: products

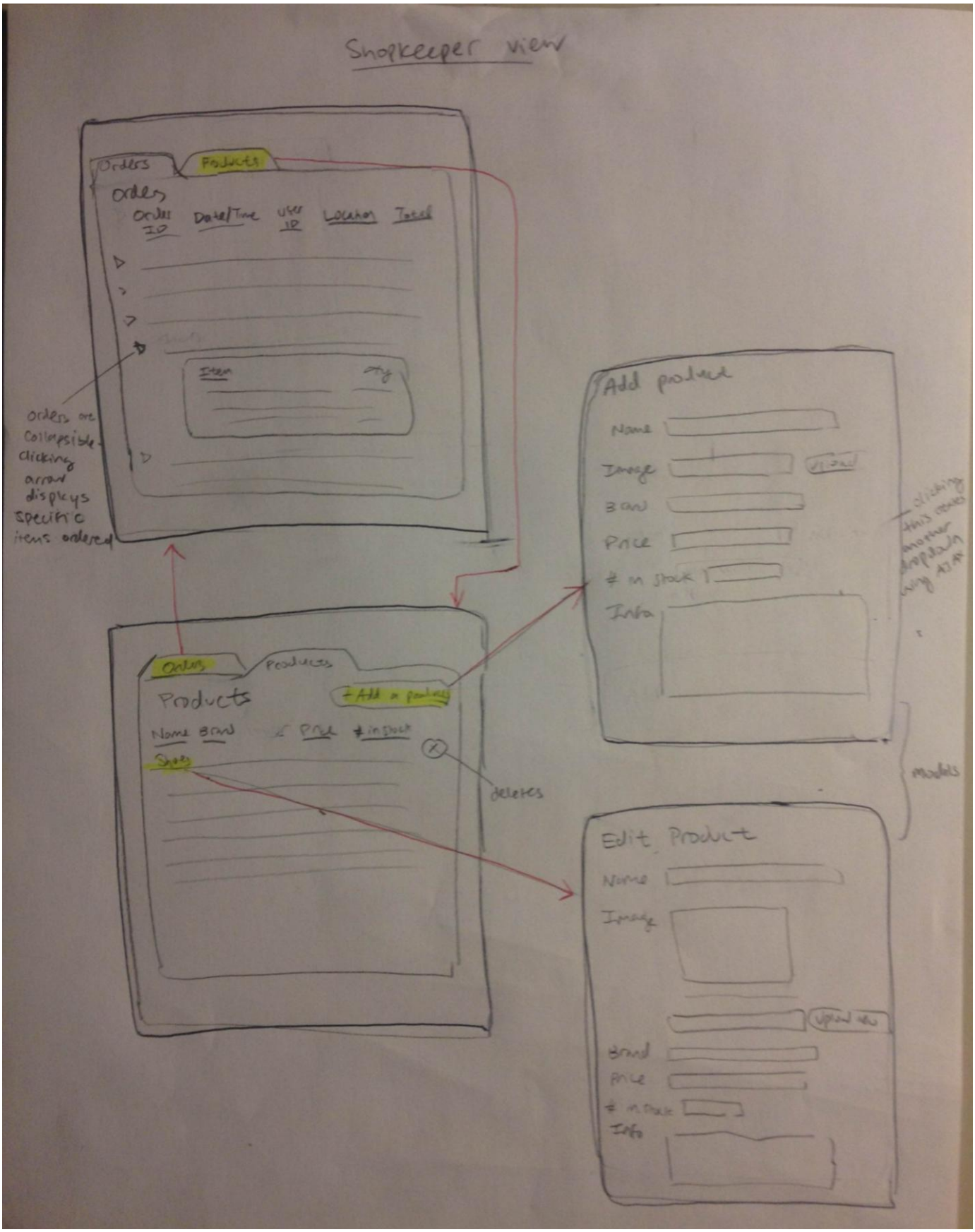
    - Effects: new product is created

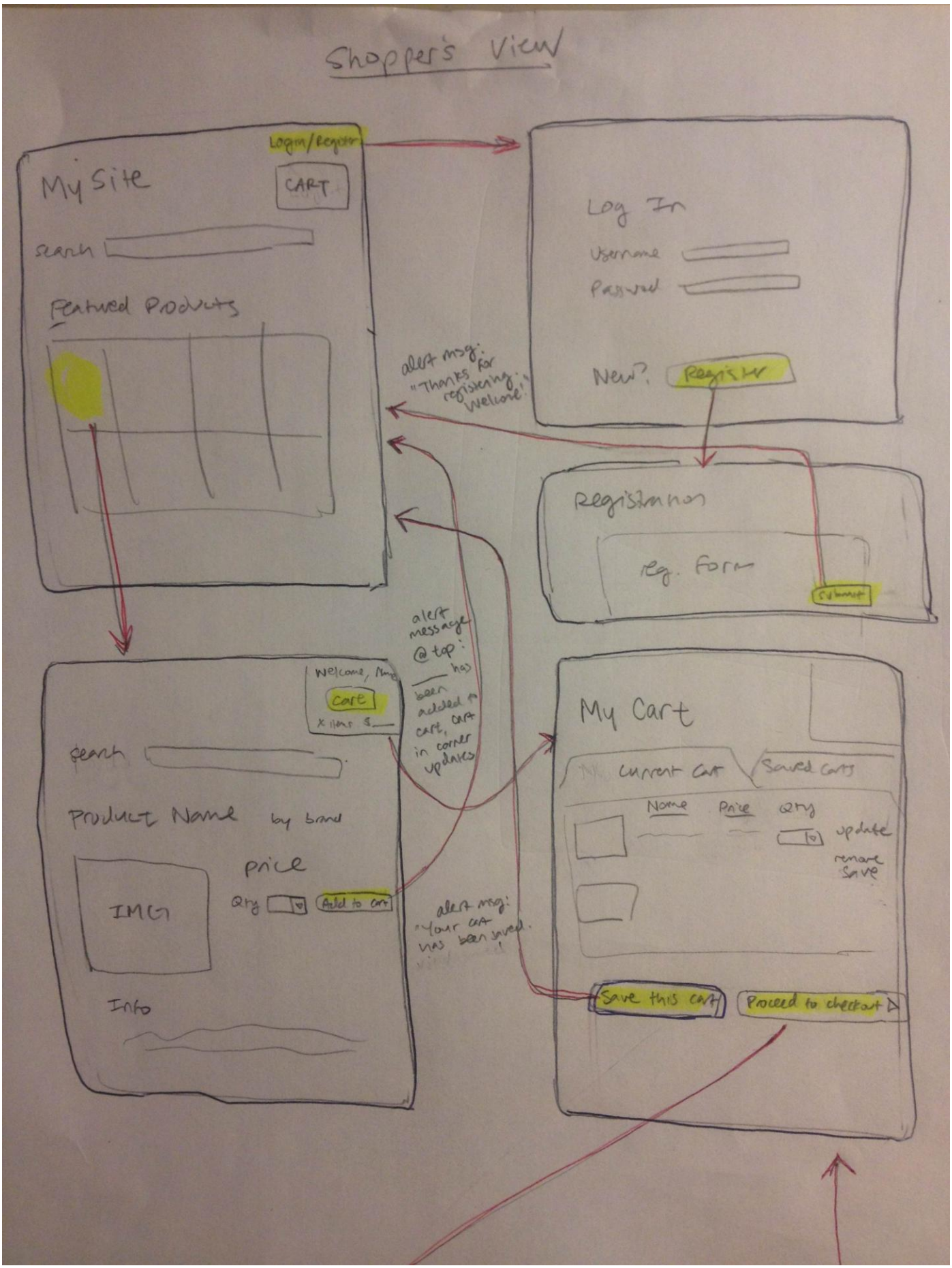
- **Edit_product** (PUT /products/id)
    - Requires: Admin is logged in, product with id exists
    - Modifies: product
    - Effects: product is updated with new attributes if validation passes

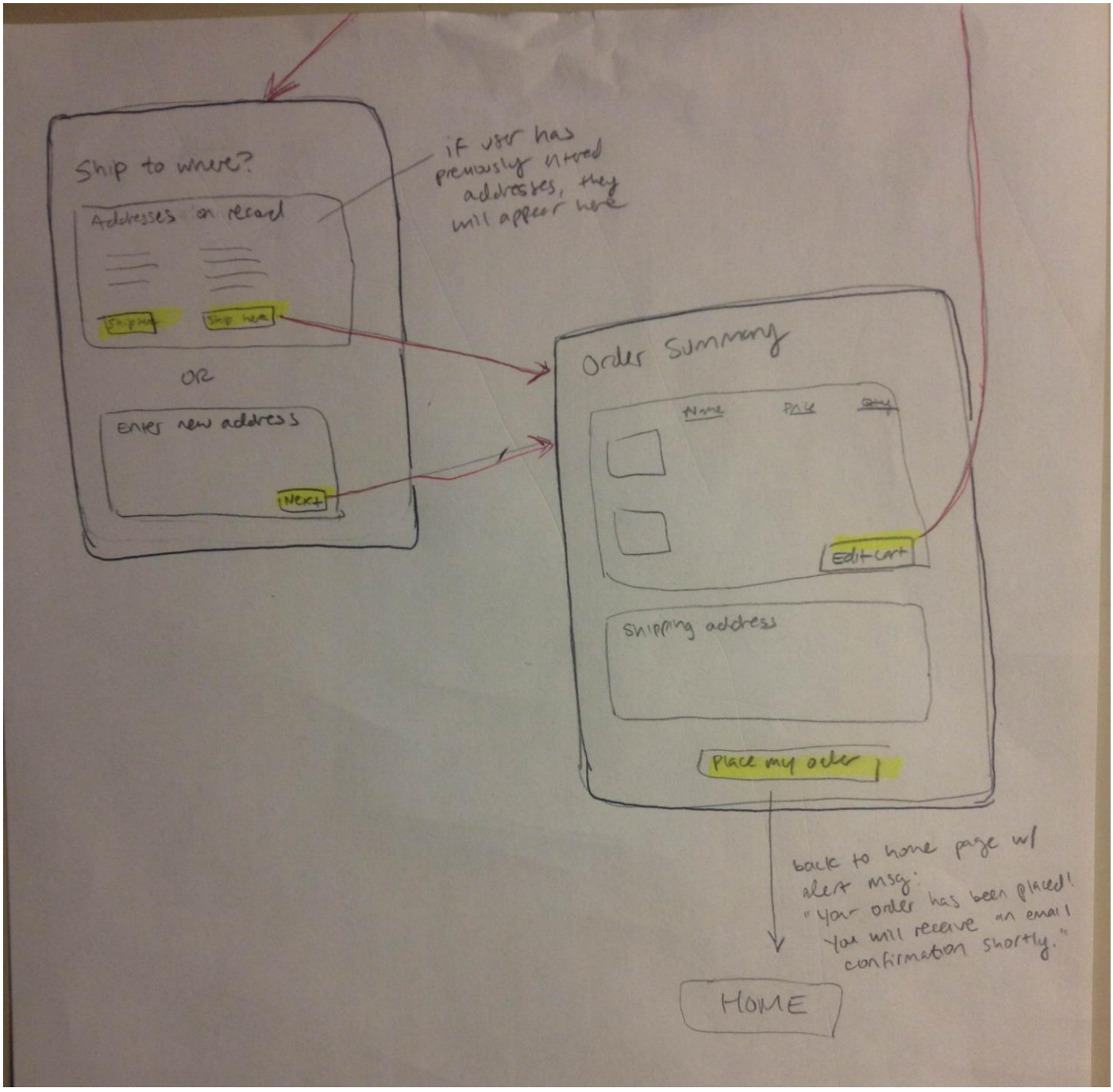
- **Remove_product** (DELETE /products/1)
    - Requires: Admin is logged in, product with id exists
    - Modifies: products
    - Effects: product is deleted from catalog

- **View_orders** (GET /admin/orders)
    - Requires: Admin is logged in

User interface


Shopkeeper view
Orders
Products
Orders
Order ID
Date/Time
User ID
Location
Total
Item
Qty
orders are collapsible - clicking arrow displays specific items ordered
Add product
Name
Image
Browse
Brand
Price
in stock
Info
clicking this opens another dropdown using AJAX
models
Orders
Products
Products
+ Add a product
Name
Brand
Price
in stock
Store
deletes
Edit Product
Name
Image
Upload new
Brand
Price
in stock
Info


Shopper's View
My Site
Login/Register
CART
search
Featured Products
Log In
Username
Password
New?
Register
alert msg:
"Thanks for registering. Welcome!"
Registration
reg. form
Welcome, Name
cart
search
Product Name
by brand
price
IMG
Qty
Add to cart
Info
alert message @ top:
has been added to cart, cart in corner updates
alert msg:
"Your cart has been saved."
My Cart
current cart
Saved carts
Name
Price
Qty
update
remove
Save
Save this cart
Proceed to checkout



(all form-filling errors result in re-direct to the same page, with errors indicated)