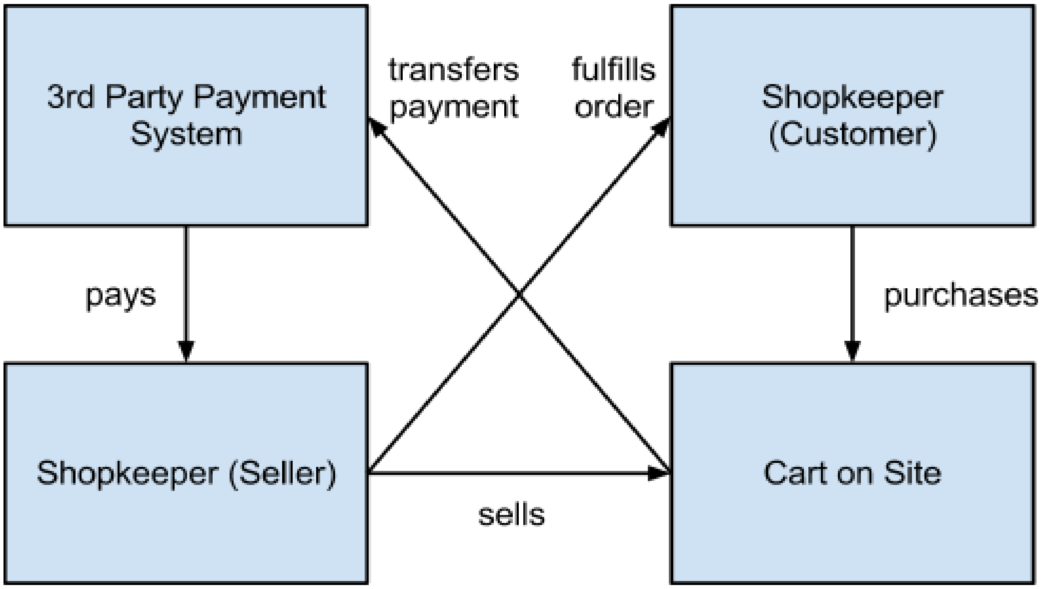
Problem Analysis PS2

Overview

Purpose and goals

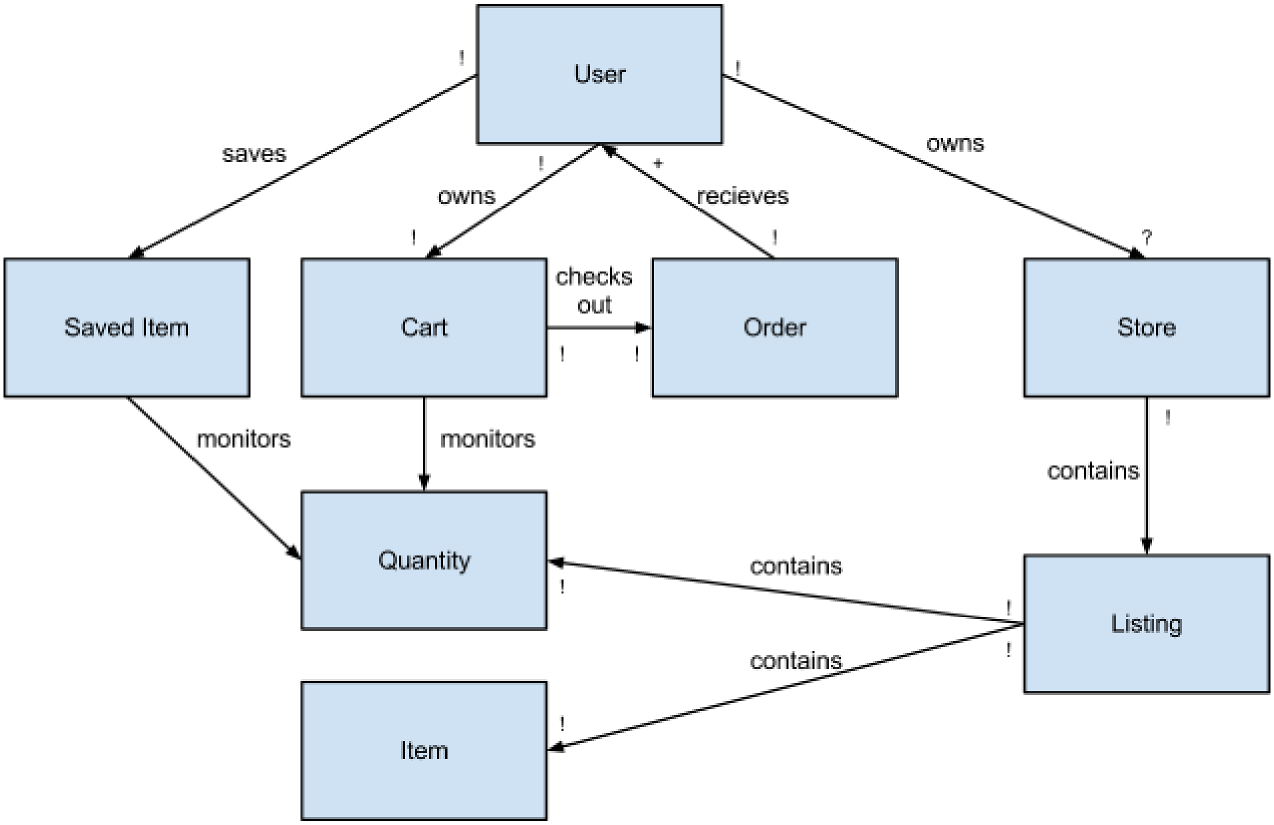
The goal of this product is to create an intuitive interface for customers and sellers to interact. This product will allow customers to easily browse and shop for products from multiple vendors. It will also be easy to shopkeepers to set up and connect with customers. Existing solutions aren't geared towards a community of customers and storekeepers, so this product will lower the barrier separating the two parties.

Context diagram



Domain

Object model



Event model
User Login
User Logout
User Create Store
User Add Item
User Edit Item: edit the price or quantity of an item
User View All Stores: view all stores, whether items are available to purchase or not
User View Store: view all items in a particular store available to purchase
User View Own Store: view all items in your own store available for editing
User View Profile: contains links to store and orders
User View Orders: view both orders placed in your store and placed by you
User Fulfill Order: view items of customer order, press fulfill
User Add to Cart: add item from particular store
User Checkout: can only be placed if items are in cart, and items available
User Save Item: save item from cart for later

ShoppingCart ::= (UserStart) (Shopper)+ (UserEnd)+
UserStart ::= Login+ ViewProfile* (CreateStore | ViewOwnStore* | (AddItem EditItem*)*)* Logout*
Shopper ::= Login* (ViewAllStores | ViewStore* | AddtoCart* SaveItem* | Checkout)* Logout*
UserEnd ::= Login+ View Orders FulfillOrder* Logout*

Behavior

Feature descriptions

- Shopping cart: shoppers can add items from any store into one cart and proceed to checkout normally
- Saved List: shoppers can save items from their cart for later, and proceed with purchase
- Shopkeeper Homepage: can easily view the status of incoming and outgoing orders, fulfill orders, edit current merchandise, and add new items
- Shopper Homepage: show all merchandise from all shopkeepers organized by stores
- Checkout: redirect to third party vendor for payment processing, confirm which items were available for purchase

Security concerns

The most sensitive part of this project is the payment procedure. In this case, all payments will be handled by a third party (think PayPal), which eliminates this risk.

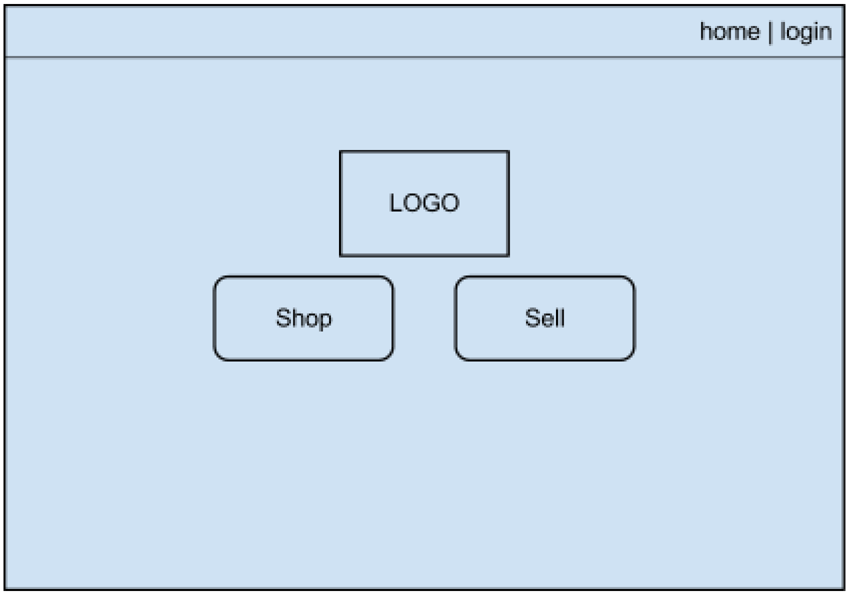
At all sensitive pages, there are checks to ensure that the user is both logged in and the “correct user,” i.e. the owner of the store if editing store items or the owner of the cart if proceeding to checkout. Sessions are hashed and also renewed at each login, to prevent attacks on this front. I also made use of attr_accessible in my models and all queries to the database are made in ways that prevent SQL-Injections. To protect from CSRF, I added the protect_from_forgery line.

Operations

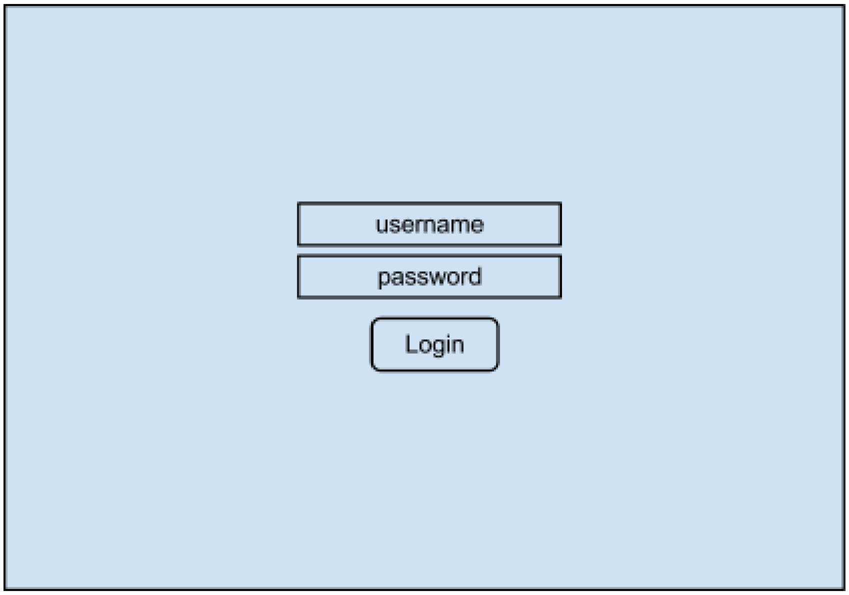
- Add to Cart
  - Pre: Item exists, Logged in
  - Frame: Item exists
  - Post: Item in Cart
- Checkout
  - Pre: Item in user cart, Item has positive quantity, Logged in
  - Frame: Item has positive quantity, Item exists
  - Post: Item not in Cart
- Fulfill Order
  - Pre: Item has pending status, Logged in
  - Frame: Item exists
  - Post: Item has fulfilled status, Order status may also change
- Create Item
  - Pre: Store exists, Logged in
  - Frame: Store exists
  - Post: Item in Store
- Create Store
  - Pre: Logged in
  - Frame: Logged in
  - Post: Store exists
- Modify Item
  - Pre: Logged in, Store exists, Item in Store

    - Frame: Item exists
    - Post: Item data updated
- View Recent Orders
    - Pre: Logged in
    - Frame: Logged in
    - Post: Logged in
- View Cart
    - Pre: Logged in
    - Frame: Logged in
    - Post: Logged in
- Save Items
    - Pre: Logged in, Item in Cart
    - Frame: Item exists
    - Post: Item is Saved, Item not in Cart
- Add Back to Cart
    - Pre: Logged in, Item in Saved
    - Frame: Item exists
    - Post: Item in Cart, Item not in Saved

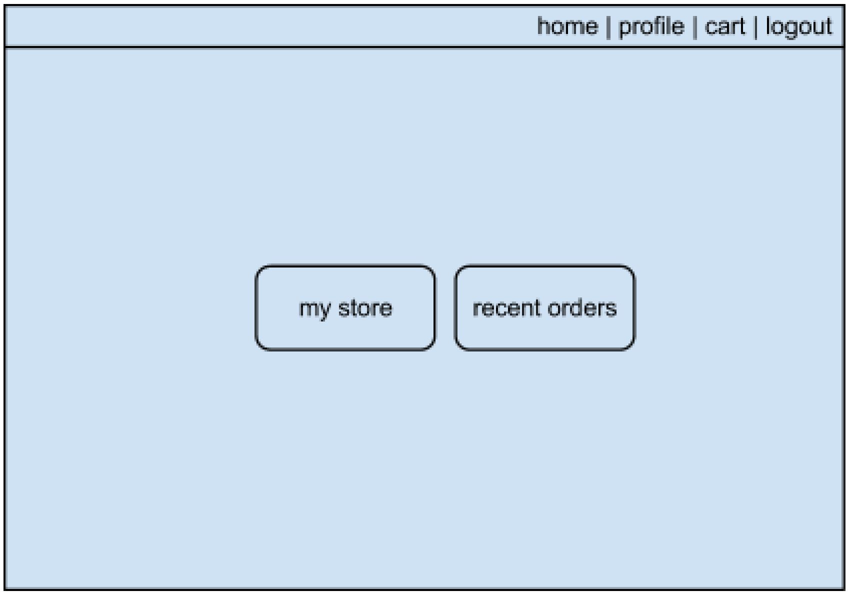
User interface



Homepage
Shop => Shopper Landing Page (After login or signup)
Sell => Storekeeper Landing Page (After login or signup)



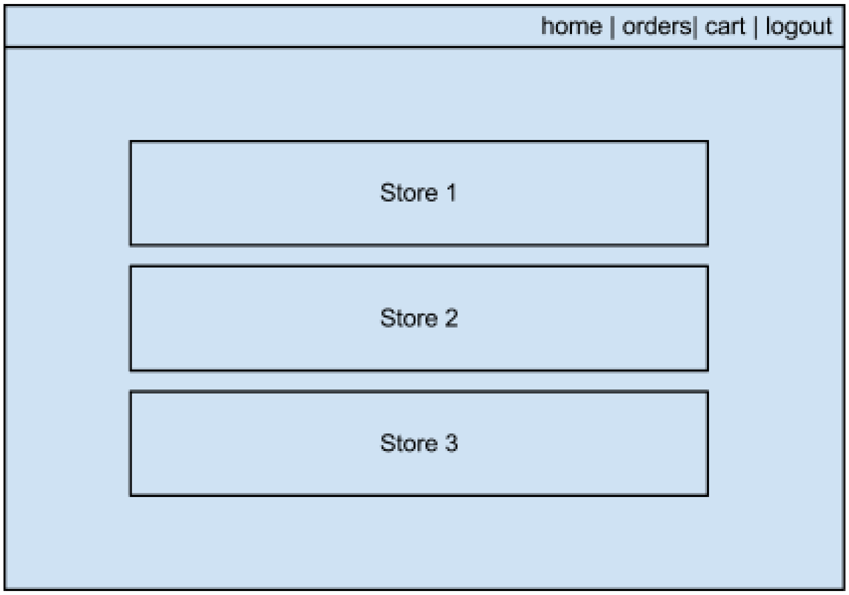
Login page (redirects to home)



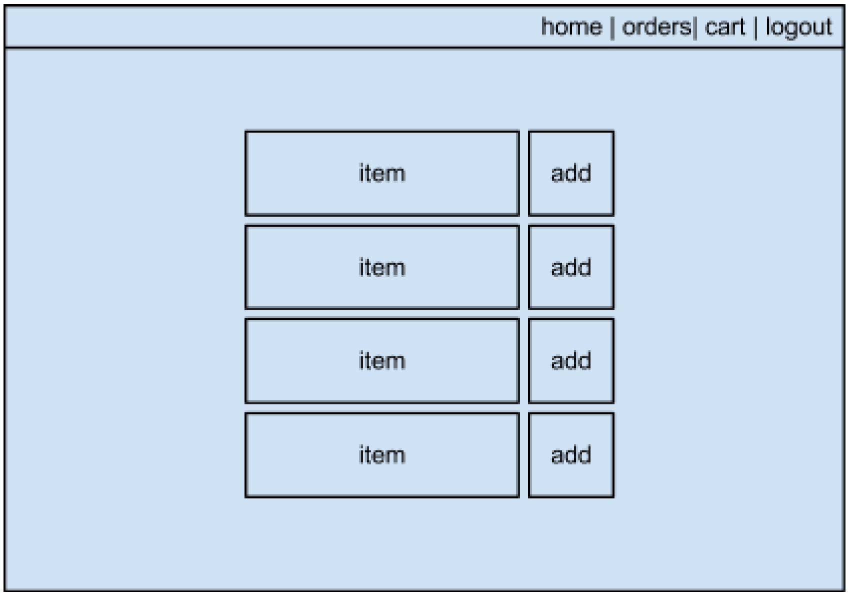
Storekeeper landing page

My store leads to page similar to shopping index page, but with edit buttons

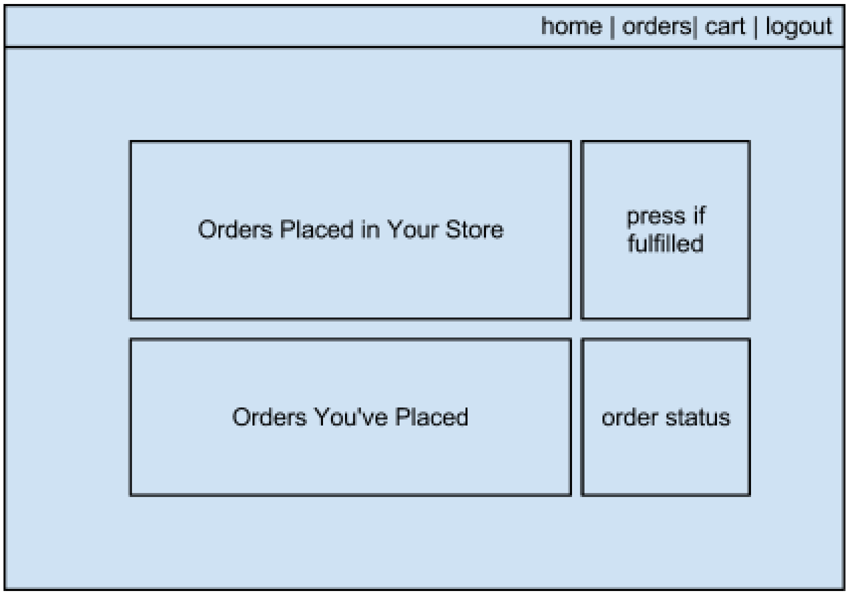
Recent orders leads to recent orders page



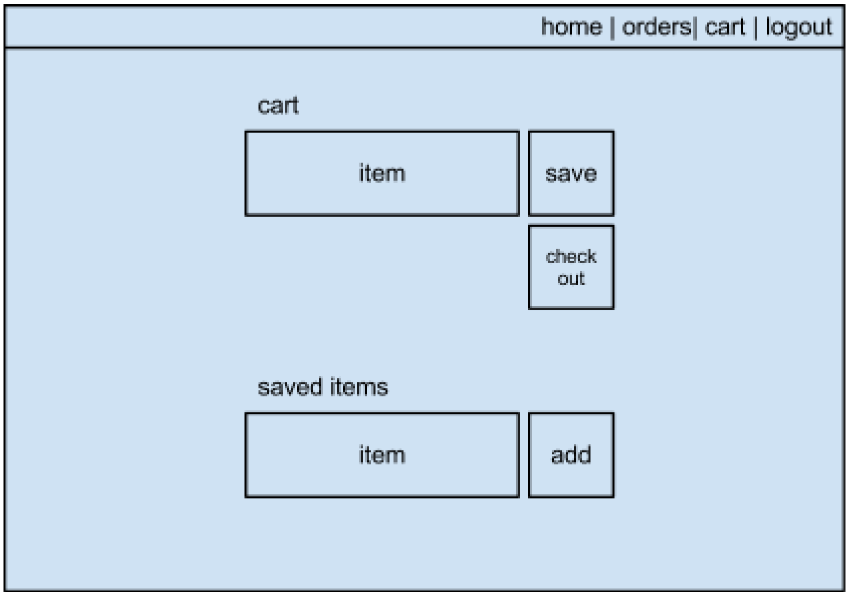
Shopper landing page
Any button in the middle screen => Shopping index page with appropriate filters



Shopping index page
Add button will add item to cart and lead to cart page



Recent orders: displays both orders you've placed and received, with the ability to fulfilled those orders placed in your store and view the status of those you've placed in other stores



Cart page: allows user to toggle items between cart and saved list, also allows for checkout