Shopping cart code review

Jonathan Edwards
10/17/2012
class Purchase < ActiveRecord::Base
  belongs_to :customer
  has_many :line_items
end

class LineItem < ActiveRecord::Base
  attr_accessible :quantity
  validates :quantity, :numericality => {:only_integer => true, :greater_than => 0}

  belongs_to :purchase
  belongs_to :product
end

class Product < ActiveRecord::Base
  belongs_to :shop
  has_many :line_items
  validates :price, :numericality => {:only_integer => true}
  validates :name, :presence => true

  attr_accessible :description, :name, :price
end
resources :purchases, :only => [:new, :create]

resources :shops, :except => [:delete]

resources :merchants, :only => [:show, :new, :create]

resources :customers, :only => [:new, :create]

match '/' => 'sessions#index', :as => :welcome, :via => :get
match '/' => 'sessions#create', :as => :login, :via => :post
match '/logout' => 'sessions#destroy', :as => :logout, :via => :delete
class PurchasesController < ApplicationController
  before_filter :require_customer

  # GET /purchases/new
  # GET /purchases/new.json

  # Shows an order review page for the user's current cart.
  def new
    @purchase = Purchase.new

    # convert the ID -> quantity form of the cart to an array
    # of [Product, quantity] pairs.
    @cart = cart.map{|id, qty| [Product.find(id), qty]}

    # Calculate the total cost of the cart.
    @total = @cart.reduce(0) do |total, pair|
      product, qty = pair
      total + (product.price * qty)
    end

    respond_to do |format|
      format.html # new.html.erb
      format.json { render json: @purchase }
    end
  end
end
%h1 Your Cart

  - if @purchase.errors.any?
    %h4 "#{pluralize(@purchase.errors.count, "error")} while checking out:"
    %ul
      - @purchase.errors.full_messages.each do |msg|
        %li= msg

%table.table.table-striped
  %thead
    %tr
      %th Item
      %th Price
      %th Quantity
  %tbody
    - @cart.each do |product, qty|
      %tr
        %td= product.name
        %td= number_to_currency (product.price / 100.0)
        %td
          .input-append
            %input.input-mini.qty-box{:type => 'text', :value => qty, 'data-price' => product.price}
            %button.btn.btn-success.cart-update-btn{:data-id => product.id} Update

%p
  %strong Total:
  %span.total_price= number_to_currency (@total / 100.0)

%p
  = link_to 'Confirm Purchase', purchases_path, :method => 'post', :class => 'btn btn-primary'
# POST /purchases
# POST /purchases.json
#
# Checks the user out: charges the user and stores the order details to report
# to merchants.
def create

respond_to do |format|
  if charged and @purchase.save
    # The charge saved and the purchase was saved. Send the user back
    # to the shops page and clear their cart.
    cookies[:cart] = nil
    format.html { redirect_to shops_url, notice: "Purchase was successful." }
    format.json { render json: @purchase, status: :created, location: @purchase }
  else
    # There was a problem. Re-render the checkout page and display
    # any errors.
    format.html { render action: "new" }
    format.json { render json: @purchase.errors, status: :unprocessable_entity }
  end
end
end
class PurchasesController < ApplicationController
  before_filter :require_customer

  # GET /purchases/new
  # GET /purchases/new.json
  #
  # Shows an order review page for the user's current cart.
  def new
    @purchase = Purchase.new

    # convert the ID -> quantity form of the cart to an array
    # of [Product, quantity] pairs.
    @cart = cart.map{|id, qty| [Product.find(id), qty]}

    # Calculate the total cost of the cart.
    @total = @cart.reduce(0) do |total, pair|
      product, qty = pair
      total + (product.price * qty)
    end
  end

  respond_to do |format|
    format.html # new.html.erb
    format.json { render json: @purchase }
  end
end
# Returns the user's cart as a map of item ID -> quantity
def cart
    (JSON.load(cookies[:cart]) || {}).reject{|k, v| v.nil? or (v < 1)}
end
helper_method :cart

# Returns the number of items in the user's cart.
def cart_size
    cart.values.inject(0, &:+)
end
helper_method :cart_size

::
class PurchasesController < ApplicationController
  before_filter :require_customer

  # GET /purchases/new
  # GET /purchases/new.json
  #
  # Shows an order review page for the user's current cart.
  def new
    @purchase = Purchase.new

    # convert the ID -> quantity form of the cart to an array
    # of [Product, quantity] pairs.
    @cart = cart.map{|id, qty| [Product.find(id), qty]}

    # Calculate the total cost of the cart.
    @total = @cart.reduce(0) do |total, pair|
      product, qty = pair
      total + (product.price * qty)
    end

    respond_to do |format|
      format.html # new.html.erb
      format.json { render json: @purchase }
    end
  end
end
# POST /purchases
# POST /purchases.json
#
# Checks the user out: charges the user and stores the order details to report
# to merchants.
def create
  @purchase = current_user.purchases.build

  # Calculate the total price and create LineItems for each item purchased
  total = 0
  cart.each do |id, qty|
    product = Product.find id
    total += (product.price * qty)

    line_item = @purchase.line_items.build :quantity => qty
    line_item.product = product
  end

  # Process payment and store the purchase details
  # Additional logic to handle payment and Persist the purchase
end