Desafío - Usando un controlador

- Para realizar este desafío debes haber estudiado previamente todo el material disponible correspondiente a la unidad.
- Una vez terminado el desafío, comprime la carpeta que contiene el desarrollo de los requerimientos solicitados y sube el .zip en el LMS.
- Puntaje total: 10 puntos.
- Desarrollo desafío:
    - El desafío se debe desarrollar de manera Individual.

Descripción

Recordando el Desafío I, dado el siguiente controlador, refactorizar el código, de tal forma que la lógica de negocio pase al modelo correspondiente.

```
class CartsController < ApplicationController
 before_action :authenticate_user!

 def update
   product = params[:cart][:product_id]
   quantity = params[:cart][:quantity]

   current_order.add_product(product, quantity)

   redirect_to root_url, notice: "Product added successfuly"
 end

 def show
   @order = current_order
 end

 def pay_with_paypal
   order = Order.find(params[:cart][:order_id])

   #price must be in cents
   price = order.total * 100

   response = EXPRESS_GATEWAY.setup_purchase(price,
     ip: request.remote_ip,
     return_url: process_paypal_payment_cart_url,
```

```
    cancel_return_url: root_url,
    allow_guest_checkout: true,
    currency: "USD"
 )

  payment_method = PaymentMethod.find_by(code: "PEC")
  Payment.create(
    order_id: order.id,
    payment_method_id: payment_method.id,
    state: "processing",
    total: order.total,
    token: response.token
 )

  redirect_to EXPRESS_GATEWAY.redirect_url_for(response.token)
end


def process_paypal_payment
  details = EXPRESS_GATEWAY.details_for(params[:token])
  express_purchase_options =
   {
      ip: request.remote_ip,
      token: params[:token],
      payer_id: details.payer_id,
      currency: "USD"
   }

  price = details.params["order_total"].to_d * 100

  response = EXPRESS_GATEWAY.purchase(price, express_purchase_options)
  if response.success?
    payment = Payment.find_by(token: response.token)
    order = payment.order

    #update object states
    payment.state = "completed"
    order.state = "completed"

    ActiveRecord::Base.transaction do
      order.save!
      payment.save!
    end
  end
```

```
  end
end
```

Asumir

- current_order siempre entrega una Order válida.

Requerimientos

1. Implementar los tests unitarios para los modelos. **(3 Puntos)**
2. Mover el código desde el controlador hacia los modelos. **(2 Puntos)**
3. Refactorizar el código. **(3 Puntos)**

Al finalizar

1. Cada método debe cumplir con el principio de Single Responsibility, incluso en el controlador. Modificar los métodos que sean necesarios. **(2 Puntos)**