

Internet y Negocio Electrónico

Departamento de Ingeniería Informática

Universidad de Cádiz

Curso 2018/2019

Informe de desarrollo de INEPlay



Grupo 1

Jesús Facio Treceño

Miguel Afán Espinosa

Carmen Ruiz de Celis

Claudia Soriano Roldán

Índice

1.1 Arquitectura Modelo - Vista - Controlador (MVC)	4
1.2 Historias de usuario	4
1.2.1 Desarrollador (Developer)	4
1.2.2 Proveedor (Supplier)	4
1.2.3 Juego (Game)	5
1.2.4 Catálogo (Catalog)	5
1.2.5 Carrito de la compra (Cart y CartItem)	5
1.2.6 Seguridad (User y UserSession)	5
1.2.7 Facturación (Order y OrderItem)	6
2.1. Introducción	7
2.2. Calendario	7
3.1. Diagrama E/R inicial	8
3.2. Diagrama E/R con funcionalidad básica	9
5.1. Sprint 1 (Generar About)	12
5.1.1. Sprint backlog y diagrama burndown	12
5.1.2. Gemas utilizadas	13
5.1.3. Ficheros de migración	13
5.1.4. Modelos ORM	13
5.1.5. Vistas y controladores	13
5.1.6. Test	15
5.1.7. Dificultades encontradas	15
5.1.8. Objetivos alcanzados	16
5.2. Sprint 2 (Gestión de Fabricantes)	17
5.2.2. Gemas utilizadas	17
5.2.3. Ficheros de migración	18
5.2.4. Modelos ORM	18
5.2.5. Vistas y controladores	19
5.2.6. Test	23
5.2.7. Dificultades encontradas	25
5.2.8. Objetivos alcanzados	25
5.3. Sprint 3 (Gestión de Proveedores)	26

5.3.2. Gemas utilizadas	26
5.3.3. Ficheros de migración	27
5.3.4. Modelos ORM	27
5.3.5. Vistas y controladores	27
5.3.6. Test	31
5.3.7. Dificultades encontradas	33
5.3.8. Objetivos alcanzados	33
5.4. Sprint 4 (Gestión de Artículos)	34
5.4.2. Gemas utilizadas	35
5.4.3. Ficheros de migración	35
5.4.4. Vistas y controladores	37
5.4.5. Test	41
5.4.6. Dificultades encontradas	45
5.4.7. Objetivos alcanzados	45
5.5. Sprint 5 (Creación del catálogo)	46
5.5.2. Gemas utilizadas	47
5.5.3. Ficheros de migración	47
5.5.4. Modelos ORM	47
5.5.5. Vistas y controladores	47
5.5.6. Test	49
5.5.7. Dificultades encontradas	52
5.5.8. Objetivos alcanzados	52
5.6. Sprint 6 (Carrito de la compra)	53
5.6.2. Gemas utilizadas	54
5.6.3. Ficheros de migración	54
5.6.4. Modelos ORM	55
5.6.5. Vistas y controladores	56
5.6.6. Test	59
5.6.7. Dificultades encontradas	60
5.6.8. Objetivos alcanzados	60
5.7. Sprint 7 (Facturación y pedidos)	61
5.7.2. Gemas utilizadas	62
5.7.3. Ficheros de migración	62
5.7.4. Modelos ORM	63
5.7.5. Vistas y controladores	66
5.7.6. Test	72
5.7.7. Dificultades encontradas	74

5.7.8. Objetivos alcanzados	74
5.6. Sprint 8 (Autenticación)	75
5.8.2. Gemas utilizadas	76
5.8.3. Ficheros de migración	76
5.8.4. Modelos ORM	77
5.8.5. Vistas y controladores	78
5.8.6. Test	91
5.8.7. Dificultades encontradas	93
5.8.8. Objetivos alcanzados	93
5.9. Sprint 9 (RSS y AJAX)	94
5.9.2. Gemas utilizadas	95
5.9.3. Ficheros de migración	95
5.9.4. Modelos ORM	95
5.9.5. Vistas y controladores	95
5.9.6. Test	100
5.9.7. Dificultades encontradas	105
5.9.8. Objetivos alcanzados	105
5.10. Sprint 10 (Borrado de órdenes)	106
5.10.3 Ficheros de migración	107
5.10.4 Modelos ORM.	107
5.10.5 Vistas y controladores	107

1.Listas de requisitos priorizados del producto (“product backlog”)

1.1 Arquitectura Modelo - Vista - Controlador (MVC)

INEplay está desarrollada con Ruby on Rails, y por lo tanto respeta la arquitectura MVC, una evolución del modelo cliente–servidor que es la más simple de todas las versiones de la arquitectura n-capas. Estas son 3 capas:

- ❖ **Modelo:** Representa la estructura de datos. Las clases que componen la capa de modelo poseen funciones de recuperación, inserción y actualización de la información almacenada en la base de datos.
- ❖ **Vista:** Es la capa más externa y más cercana al usuario. Es la responsable de generar una interfaz de usuario, usualmente basada en los datos del modelo. Aunque pueda presentar varias formas de introducción de datos, ésta nunca gestiona los datos entrantes.
- ❖ **Controlador:** Es la capa que actúa de intermediaria entre las vistas y los modelos, ayudando a procesar y dirigir las peticiones de usuario y generando las páginas pertinentes. Es decir, es la responsable de orquestar la aplicación.

1.2 Historias de usuario

1.2.1 Desarrollador (Developer)

- ❖ Añadir desarrollador: dado un nuevo desarrollador, el encargado o dueño de la tienda será el responsable de darlo de alta en el sitio. Se deben proporcionar los datos pedidos en el formulario. Una vez enviado, los datos del nuevo proveedor estarán disponibles para usar.
- ❖ Listar desarrolladores: página que sirve como lista de todos los desarrolladores incluidos en el sitio.
- ❖ Editar desarrollador: puede surgir la necesidad de modificar los datos de un proveedor ya existente y debe poder realizarse de forma ágil y rápida.
- ❖ Eliminar desarrollador: los desarrolladores deben poder ser dados de baja y eliminados del sistema.

1.2.2 Proveedor (Supplier)

- ❖ Añadir proveedor: dado un nuevo proveedor, el encargado o dueño de la tienda será el responsable de darlo de alta en el sitio. Se deben proporcionar los datos pedidos en el formulario. Una vez enviado, los datos del nuevo proveedor estarán disponibles para usar.
- ❖ Listar proveedores: página que sirve como lista de todos los proveedores incluidos en el sitio.

- ❖ Editar proveedor: puede surgir la necesidad de modificar los datos de un proveedor ya existente y debe poder realizarse de forma ágil y rápida.
- ❖ Eliminar proveedor: los proveedores deben poder ser dados de baja y eliminados del sitio.

1.2.3 Juego (Game)

- ❖ Añadir juego: dado un nuevo juego, el encargado o dueño de la tienda será el responsable de darlo de alta en el sitio. Se deben proporcionar los datos pedidos en el formulario. Una vez enviado, los datos del nuevo juego estarán disponibles para usar.
- ❖ Listar juegos: página que sirve como lista de todos los desarrolladores incluidos en el sitio.
- ❖ Editar juego: puede surgir la necesidad de modificar los datos de un juego ya existente y debe poder realizarse de forma ágil y rápida.
- ❖ Eliminar juego: los juegos deben poder ser dados de baja y eliminados del sitio.

1.2.4 Catálogo (Catalog)

- ❖ Ver catálogo: el usuario debe poder consultar el catálogo de artículos, que mostrará todo lo que se puede comprar en la tienda online.
- ❖ Mostrar artículo del catálogo: un usuario debe poder acceder a los detalles de un determinado artículo publicado en el catálogo.

1.2.5 Carrito de la compra (Cart y CartItem)

- ❖ Añadir: Un cliente tendrá a su disposición un carrito de la compra virtual donde puede insertar los productos que deseé comprar.
- ❖ Eliminar: De la misma forma que un usuario puede insertar artículos en el carrito, también debe poder eliminar los artículos del carrito uno a uno, vaciar el carrito entero.

Estas operaciones serán accesibles mediante enlaces claros e intuitivos. Las operaciones serán sencillas de realizar y amigables con el usuario.

1.2.6 Seguridad (User y UserSession)

- ❖ Inicio de sesión correcto: cuando un usuario que no ha iniciado sesión intent realizar alguna función que requiera de autentificación, el sistema le redirigirá a la página de inicio de sesión pertinente. Una vez redirigido, si las credenciales introducidas por el usuario son correctas, tras iniciar sesión el usuario será devuelto a la página en la que estaba anteriormente para poder continuar realizando la acción que deseaba.
- ❖ Inicio de sesión incorrecto: cuando un usuario que no ha iniciado sesión intent realizar alguna función que requiera de autentificación, el sistema le redirigirá a la página de inicio de sesión pertinente. Una vez redirigido, si las credenciales introducidas por el usuario son incorrectas, el sistema debe notificar al usuario del error.

1.2.7 Facturación (Order y OrderItem)

- ❖ Facturar: cuando el cliente haya añadido todos los artículos que deseé al carrito de la compra, podrá proceder a la página de facturación, donde se le requerirán datos como la dirección de envío y la información de la tarjeta de crédito.
Una vez introducidos los datos, se procede a la emisión y al procesamiento del pedido.
- ❖ Listar pedidos: el administrador debe poder listar el estado de todos los pedidos del sistema y debe poder obtener información del estado de cada uno. Un pedido procesado es aquel que ha sido cobrado al cliente pero aún no ha sido enviado. Un pedido cerrado es aquel que ya ha sido cobrado y enviado.
- ❖ Ver pedido: antes de enviar el pedido, el administrador debe poder ver los detalles del pedido. Esta página mostrará la dirección de envío y los datos de facturación junto con la información del cliente.
- ❖ Cerrar pedidos: Un pedido facturado y enviado debe pasar a “cerrado”. Esta acción se realizará con la simple pulsación de un botón.
- ❖ Borrar pedidos : el administrador debe poder borrar los pedidos.

2. Planificación del proyecto

2.1. Introducción

Para planificar el proyecto, nos hemos basado en SCRUM, una metodología de desarrollo ágil. Para representar las etapas del desarrollo hemos usado los sprints.

2.2. Calendario

CRONOGRAMA DEL PROYECTO

Explicación →

Este cronograma es una variante del gráfico de Gantt, y consiste en crear una programación de un proyecto desglosada en fases.

TÍTULO	INEPlay	NOMBRE DE LA EMPRESA	UCA
FECHA	23/05/19		



3. Esquema de la base de datos

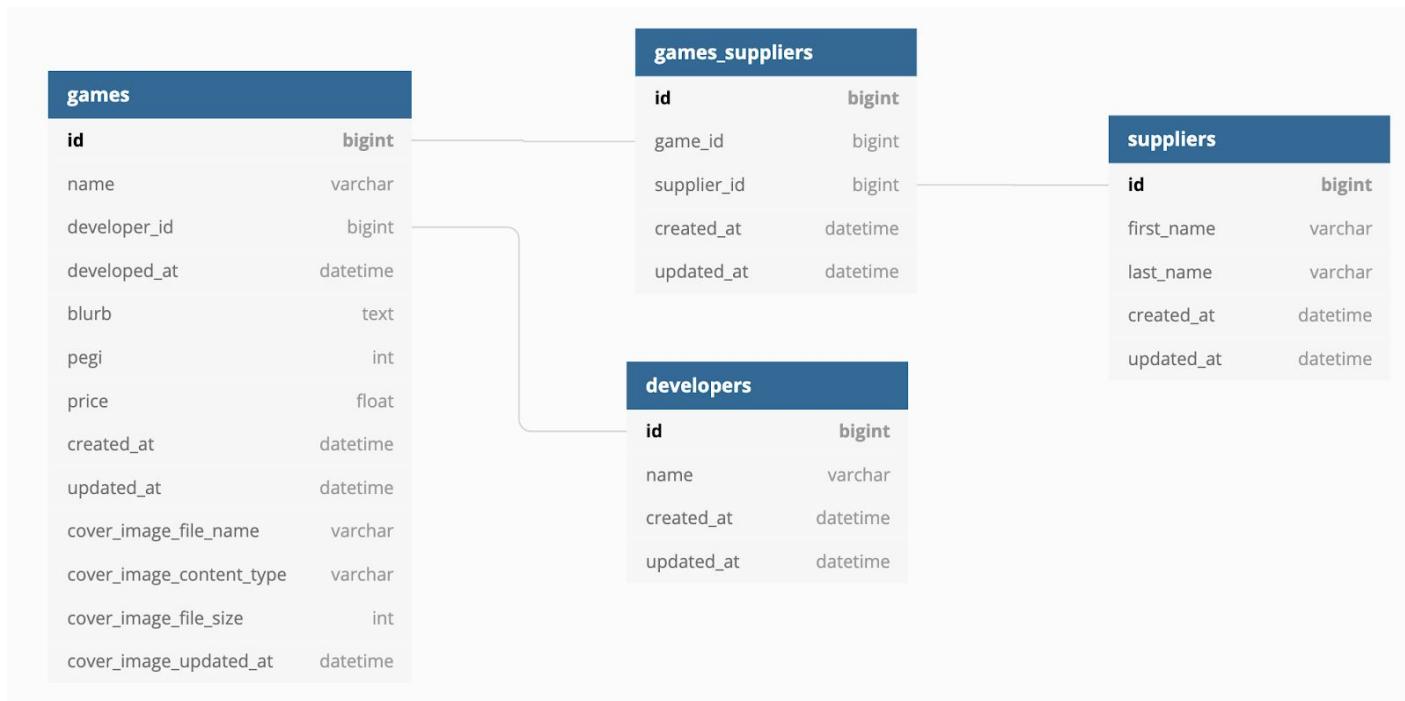
Aquí observamos la evolución del diagrama E-R que representa el esquema básico de la Base de Datos del sitio.

3.1. Diagrama E/R inicial

En los primeros sprints trabajamos con la entidad principal (`games`), las dos entidades secundarias (`suppliers` y `developers`) y la entidad relación (`games_suppliers`).

Por lo tanto nuestro esquema inicial de la base de datos contendría:

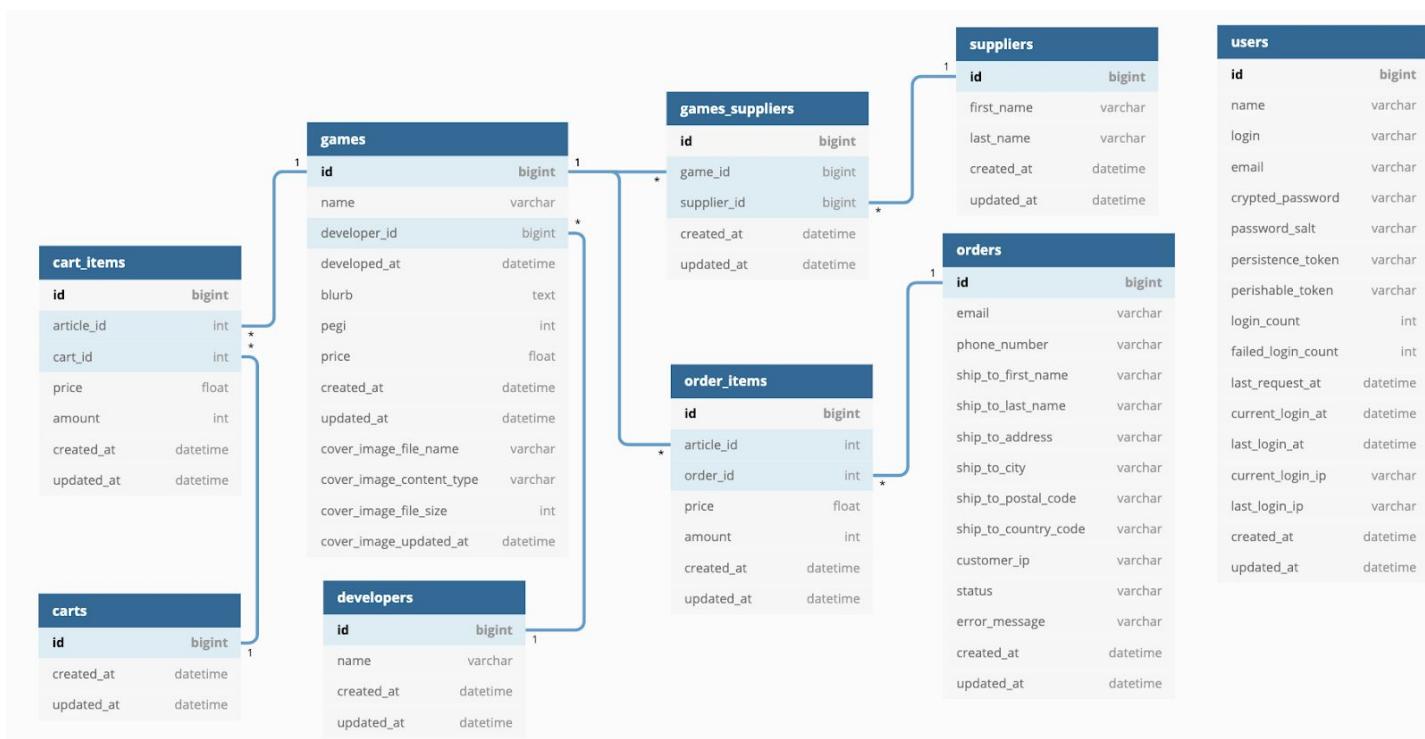
- ❖ Games
- ❖ Developers
- ❖ Suppliers



3.2. Diagrama E/R con funcionalidad básica

Hasta la octava iteración hemos implementado todas las funcionalidades básicas, incluyendo el carrito de la compra, los pedidos y la seguridad. El nuevo diagrama contaría con las siguientes entidades:

- ❖ Games
- ❖ Developers
- ❖ Suppliers
- ❖ Carts
- ❖ Orders
- ❖ Users



4. Contenido del fichero de rutas

Finalmente, el fichero de rutas resulta así:

Fichero: config / routes.rb

```
Rails.application.routes.draw do

  root :to => 'catalog#index'

  get 'about' => 'about#index'
  get 'catalog' => 'catalog#index'
  get 'checkout' => 'checkout#index'
  get 'admin/developer' => 'admin/developer#index'
  get 'admin/supplier' => 'admin/supplier#index'
  get 'admin/game' => 'admin/game#index'
  get 'admin/order' => 'admin/order#index'

  get 'about/index'

  get 'admin/developer/new'
  post 'admin/developer/create'
  get 'admin/developer/edit'
  post 'admin/developer/update'
  post 'admin/developer/destroy'
  get 'admin/developer/show'
  get 'admin/developer/show/:id' => 'admin/developer#show'
  get 'admin/developer/index'

  get 'admin/supplier/new'
  post 'admin/supplier/create'
  get 'admin/supplier/edit'
  post 'admin/supplier/update'
  post 'admin/supplier/destroy'
  get 'admin/supplier/show'
  get 'admin/supplier/show/:id' => 'admin/supplier#show'
  get 'admin/supplier/index'

  get 'admin/game/new'
  post 'admin/game/create'
  get 'admin/game/edit'
  post 'admin/game/update'
  post 'admin/game/destroy'
  get 'admin/game/show'
  get 'admin/game/show/:id' => 'admin/game#show'
  get 'admin/game/index'

  post 'admin/order/close'
  post 'admin/order/destroy'
  get 'admin/order/show'
  get 'admin/order/show/:id' => 'admin/order#show'
  get 'admin/order/index'
```

```
get 'catalog/show'
get 'catalog/show/:id' => 'catalog#show'
get 'catalog/index'
get 'catalog/latest'
get 'catalog/rss'

get 'cart/add'
post 'cart/add'
get 'cart/remove'
post 'cart/remove'
get 'cart/clear'
post 'cart/clear'

get 'user_sessions/new'
get 'user_sessions/create' # for showing failed login screen after restarting web server
post 'user_sessions/create'
get 'user_sessions/destroy'

get 'user/new'
post 'user/create'
get 'user/show'
get 'user/show/:id' => 'user#show'
get 'user/edit'
post 'user/update'

get 'checkout/index'
post 'checkout/submit_order'
get 'checkout/thank_you'

get 'password_reset/new'
post 'password_reset/create'
get 'password_reset/edit'
post 'password_reset/update'

# For details on the DSL available within this file, see http://guides.rubyonrails.org/routing.html
end
```

5. Sprints

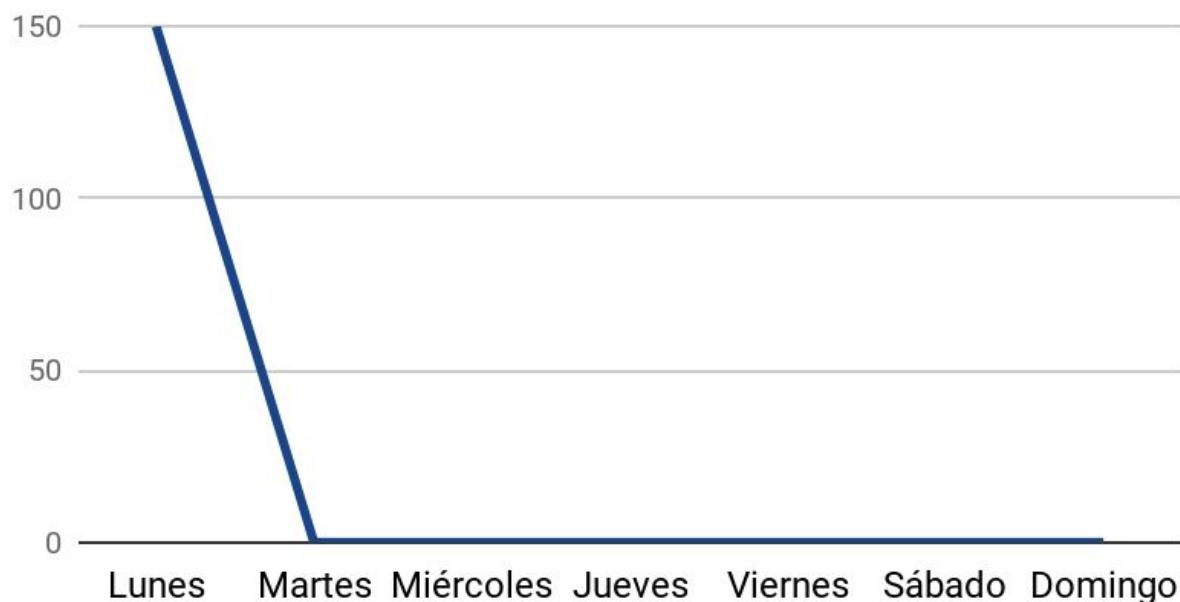
5.1. Sprint 1 (Generar About)

En esta iteración lo que hemos hecho es generar el about de nuestra página web y aprender los fundamentos del framework Ruby on Rails.

5.1.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	0	0	0	0	0	0	0
Vista y Controlador	60	0	0	0	0	0	0
Modelo	90	0	0	0	0	0	0
Test	0	0	0	0	0	0	0
Total	150	0	0	0	0	0	0

Sprint 1



5.1.2. Gemas utilizadas

- rails , versión 4.2.5: Framework de Ruby on Rails.
- mysql2 , versión 0.4.2: Controlador de MySQL.
- sass-rails, versión 5.0.6: Utiliza SCSS para la hoja de estilos.
- uglifier, versión 2.7.2: Comprime los archivos de JavaScript.
- coffee-rails, versión 4.1.1: Permite utilizar CoffeeScript en Rails.
- turbolinks, versión 2.5.3: Acelera el seguimiento de enlaces.
- jbuilder, versión 2.3.2: Construye JSON APIs con facilidad.
- sdoc, versión 0.4.1 : Genera documentación de API.
- spring, versión 1.6.1: Permite mantener la aplicación en segundo plano durante el desarrollo.

5.1.3. Ficheros de migración

En este sprint no hemos añadido ficheros de migración en db/migrate

5.1.4. Modelos ORM

En este sprint no ha sido necesario crear ningún modelo

5.1.5. Vistas y controladores

Hemos modificado el fichero del controlador, para que se muestre el título de la página:

Fichero: app / controllers / about_controller.rb

```
class AboutController < ApplicationController
  def index
    @page_title = 'Sobre INEPlay'
  end
end
```

Después añadimos la hoja de estilo del proyecto de ejemplo Minitemporium de app/assets/stylesheets/style.css y modificamos el archivo, que quedaría de la siguiente manera:

Fichero: app / views / layouts / application.html.erb

```

<!DOCTYPE html>
<html>
<head>
  <title><%= @page_title || 'INEPlay' %></title>
  <%= csrf_meta_tags %>
  <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track': 'reload' %>
  <%= javascript_include_tag 'application', 'data-turbolinks-track': 'reload' %>
</head>

<body>
  <div id="header">
    <h1 id="logo">INEPlay&trade;</h1>
    <h2 id="slogan">Videojuegos sobre Railes</h2>
  </div>

  <div id="menu">
    <ul>
      <li><a href="/admin/developer">Desarrolladores</a>&ampnbsp|&ampnbsp</li>
      <li><a href="/admin/supplier">Proveedores</a>&ampnbsp|&ampnbsp</li>
      <li><a href="/admin/game">Juegos</a>&ampnbsp|&ampnbsp</li>
      <li><a href="/admin/order">Pedidos</a>&ampnbsp|&ampnbsp</li>
      <li><a href="/">Catálogo</a>&ampnbsp|&ampnbsp</li>
      <li><a href="/about">Sobre INEPlay</a>&ampnbsp</li>
    </ul>
  </div>

  <div id="content">
    <h1><%= @page_title if @page_title %></h1>
    <% if flash[:notice] %>
      <div id="notice"><%= flash[:notice] %></div>
    <% end %>
    <%= yield %>
  </div>

  <% if @cart %>
    <div id="shopping_cart"><%= render :partial => 'cart/cart' %></div>
  <% end %>

  <div id="footer">
    &copy; 2015-2019 INEPlay
  </div>
</body>
</html>

```

Modificamos la vista `about#index`, desde su fichero de vista.

Fichero: `app / views / about / index . html . erb`

```
<p> Tienda de videojuegos online de Puerto Real, Cádiz</p>
<h2>Dirección postal</h2>
<address>
INEPlay<br/>
C.P. 11519<br/>
ESI<br/>
</address>
```

5.1.6. Test

En esta ocasión hemos hecho un test de controlador.

Fichero: test / controllers / about_controller_test.rb

```
require 'test_helper'

class AboutControllerTest < ActionDispatch::IntegrationTest
  test "index" do
    get '/about/index'
    assert_response :success
    assert_template 'about/index'
    assert_equal 'Sobre INEPlay', assigns(:page_title)
    assert_select 'title', 'Sobre INEPlay'
  end
end
```

5.1.7. Dificultades encontradas

Para todos los integrantes de nuestro grupo, Ruby on Rails es algo novedoso. Tuvimos dificultades con su instalación, la instalación de las gemas y la integración de la base de datos. Estas dificultades fueron solventadas con ayuda del profesor en clase y consultando el libro de Beginning Ruby on Rails ECommerce.

Además el profesor nos proporcionó un código (Miniemporium) que nos sirvió de referencia a lo largo de todo el proceso de creación de nuestra tienda online.

5.1.8. Objetivos alcanzados

En este sprint hemos conseguido:

- Aprender los fundamentos básicos del funcionamiento de Ruby y Ruby on Rails.
- Generar y visualizar correctamente el about de nuestra tienda online.

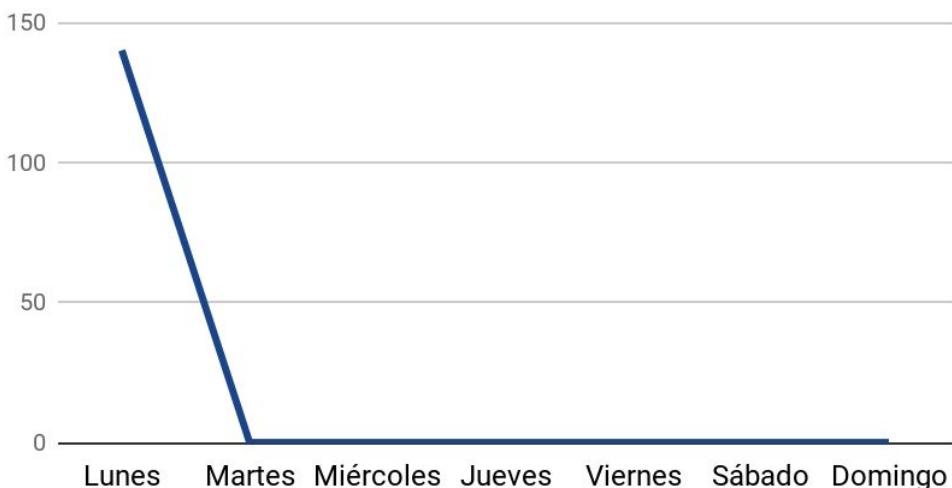
5.2. Sprint 2 (Gestión de Fabricantes)

En esta iteración el objetivo ha sido generar la entidad secundaria Developer, que corresponde a los fabricantes (desarrolladores) de la entidad principal Game.

5.2.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	20	0	0	0	0	0	0
Vista y Controlador	60	0	0	0	0	0	0
Modelo	40	0	0	0	0	0	0
Test	20	0	0	0	0	0	0
Total	140	0	0	0	0	0	0

Sprint 2



5.2.2. Gemas utilizadas

En esta iteración no hemos utilizado gemas nuevas.

5.2.3. Ficheros de migración

El comando que permite crear las plantillas del modelo, del test del modelo y del fichero de migración de Developer es el siguiente:

```
$ rails generate model Developer
```

Modificamos el fichero de migración para generar la estructura de la tabla:

Fichero: \db\migrate\20190506120001_create_developers.rb

```
class CreateDevelopers < ActiveRecord::Migration[5.1]
  def change
    create_table :developers do |t|
      t.string :name, :limit => 255, :null => false, :unique => true
      t.timestamps
    end
  end
end
```

Para que se creen convenientemente las tablas en la base de datos:

```
ine@gii ~/projects/eshop1 $ rake db:migrate
```

5.2.4. Modelos ORM

Modificamos el fichero del modelo:

Fichero: app / models / developer.rb

```
class Developer < ApplicationRecord
  has_many :games
  validates_presence_of :name, :message => '(Nombre) no puede dejarse en blanco'
  validates_uniqueness_of :name, :message => '(Nombre) no aceptado'
  validates_length_of :name, :in => 2..255, :message => '(Nombre) es muy corto, debe contener al menos dos letras'
end
```

5.2.5. Vistas y controladores

Generamos el controlador manufacturer, dentro de admin:

```
ine@gii ~/projects/eshop1 $ rails generate controller admin/developer new  
create edit update destroy show index
```

A continuación, modificamos el fichero del controlador:

Fichero: app / controllers / admin / developer_controller.rb

```
class Admin::DeveloperController < ApplicationController  
  def new  
    @developer = Developer.new  
    @page_title = 'Crear nuevo desarrollador'  
  end  
  
  def create  
    @developer = Developer.new(developer_params)  
    if @developer.save  
      flash[:notice] = "Desarrollador #{@developer.name} creado correctamente."  
      redirect_to :action => 'index'  
    else  
      @page_title = 'Crear nuevo desarrollador'  
      render :action => 'new'  
    end  
  end  
  
  def edit  
    @developer = Developer.find(params[:id])  
    @page_title = 'Editar desarrollador'  
  end  
  
  def update  
    @developer = Developer.find(params[:id])  
    if @developer.update_attributes(developer_params)  
      flash[:notice] = "Desarrollador #{@developer.name}  
actualizada correctamente."  
      redirect_to :action => 'show', :id => @developer  
    else  
      @page_title = 'Editar desarrollador'  
      render :action => 'edit'  
    end  
  end  
end
```

```

def show
  @developer = Developer.find(params[:id])
  @page_title = @developer.name
end

def index
  @developers = Developer.all
  @page_title = 'Listado de desarrolladores'
end

def destroy
  @developer = Developer.find(params[:id])
  @developer.destroy
  flash[:notice] = "Desarrollador #{@developer.name} eliminado correctamente."
  redirect_to :action => 'index'
end

private
  def developer_params
    params.require(:developer).permit(:name)
  end

```

Y seguidamente los ficheros correspondientes a las vistas:

Fichero: app / views / admin / developer / form.html.erb

```

<% if @developer.errors.any? %>
  <div id="errorExplanation">
    <h2>Este desarrollador no se puede guardar debido a <%= pluralize(@developer.errors.count, "error", "errores") %></h2>
    <ul>
      <% @developer.errors.full_messages.each do |msg| %>
        <li><%= msg %></li>
      <% end %>
    </ul>
  </div>
<% end %>

<div class="field">
  <p><label for="developer_name">Nombre</label><br/>
  <%= text_field 'developer', 'name' %></p>
</div>

```

Fichero: app / views / admin / developer / developer.html.erb

```
<tr>
  <td><%= link_to developer.name, :action => 'show', :id => developer %></td>
  <td><%= link_to 'Editar', :action => 'edit', :id => developer %></td>
  <td>
    <%= button_to 'Eliminar', { :action => 'destroy', :id => developer },
      data: { confirm: "¿Está seguro de que quiere eliminar el desarrollador #{developer.name}?" } %>
  </td>
</tr>
```

Fichero: app / views / admin / developer / edit.html.erb

```
<%= form_tag :action => 'update', :id => @developer do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Actualizar Desarrollador' %>
<% end %>

<%= link_to 'Mostrar', :action => 'show', :id => @developer %> |
<%= link_to 'Atrás', :action => 'index' %>
```

Fichero: app / views / admin / developer / index.html.erb

```
<table>
  <tr>
    <th>Nombre</th>
    <th>Editar</th>
    <th>Eliminar</th>
  </tr>

  <%= render :partial => 'developer', :collection => @developers %>
</table>

<p><%= link_to 'Añadir un nuevo desarrollador', :action => 'new' %></p>
```

Fichero: app / views / admin / developer / new.html.erb

```
<%= form_tag :action => 'create' do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Crear Desarrollador' %>
<% end %>

<%= link_to 'Atrás', :action => 'index' %>
```

Fichero: app / views / admin / developer / show.html.erb

```
<dl>
  <dt>Nombre</dt>
  <dd><%= @developer.name %></dd>
</dl>

<%= link_to 'Editar', :action => 'edit', :id => @developer %> |
<%= link_to 'Atrás', :action => 'index' %>
```

5.2.6. Test

Vamos a preparar el fichero de tests, en primer lugar el del modelo:

Fichero: test / models / developer_test . rb

```
require 'test_helper'

class DeveloperTest < ActiveSupport::TestCase
  test "failing_create" do
    developer = Developer.new
    assert_equal false, developer.save
    assert_equal 2, developer.errors.count
    assert developer.errors[:name]

  end

  test "create" do
    developer = Developer.new(
      :name => 'Atari',
      )

    assert developer.save
  end
end
```

Y a continuación el del controlador:

Fichero: test / controllers / admin / developer_controller_test . rb

```
require 'test_helper'

class Admin::DeveloperControllerTest < ActionDispatch::IntegrationTest
  fixtures :developers

  test "new" do
    get '/admin/developer/new'
    assert_response :success
  end

  test "create" do
    num_developers = Developer.count
    post '/admin/developer/create', :params => { :developer => { :name => 'The Monopoly Developing Company' } }
    assert_response :redirect
    assert_redirected_to :action => 'index'
    assert_equal num_developers + 1, Developer.count
  end
```

```

test "edit" do
  get '/admin/developer/edit', :params => { :id => 1 }
  assert_select 'input' do
    assert_select '[type=?]', 'text'
    assert_select '[name=?]', 'developer[name]'
    assert_select '[value=?]', 'ubisoft'
  end
end

test "update" do
  post '/admin/developer/update', :params => { :id => 1, :developer => { :name => 'ubisoft' } }
  assert_response :redirect
  assert_redirected_to :action => 'show', :id => 1
  assert_equal 'ubisoft', Developer.find(1).name
end

test "destroy" do
  assert_difference(Developer, :count, -1) do
    post '/admin/developer/destroy', :params => { :id => 1 }
    assert_equal flash[:notice], 'Desarrollador ubisoft eliminado correctamente.'
    assert_response :redirect
    assert_redirected_to :action => 'index'
    get '/admin/developer/index'
    assert_response :success
    assert_select 'div#notice', 'Desarrollador ubisoft eliminado correctamente.'
  end
end

test "show" do
  get '/admin/developer/show', :params => { :id => 1 }
  assert_response :success
  assert_template 'admin/developer/show'
  assert_not_nil assigns(:developer)
  assert assigns(:developer).valid?
  assert_select 'div#content' do
    assert_select 'h1', Developer.find(1).name
  end
end

test "index" do
  get '/admin/developer/index'
  assert_response :success
  assert_select 'table' do
    assert_select 'tr', Developer.count + 1
  end
  Developer.find_each do |a|
    assert_select 'td', a.name
  end
end

```

Y luego ejecutamos los tests:

```
rake test TEST=test/models/developer_test.rb
```

5.2.7. Dificultades encontradas

Gracias al código de Minitemporium pudimos tener referencias y realizar la mayor parte del trabajo sin problemas, aún así al no estar acostumbrados a trabajar con Ruby on Rails volvimos a tener problemas con ciertas gemas y con las migraciones de la base de datos. Usamos los comandos “bundle install ” y “rake db: migrate”.

5.2.8. Objetivos alcanzados

Visualizar el CRUD de la entidad Developer.

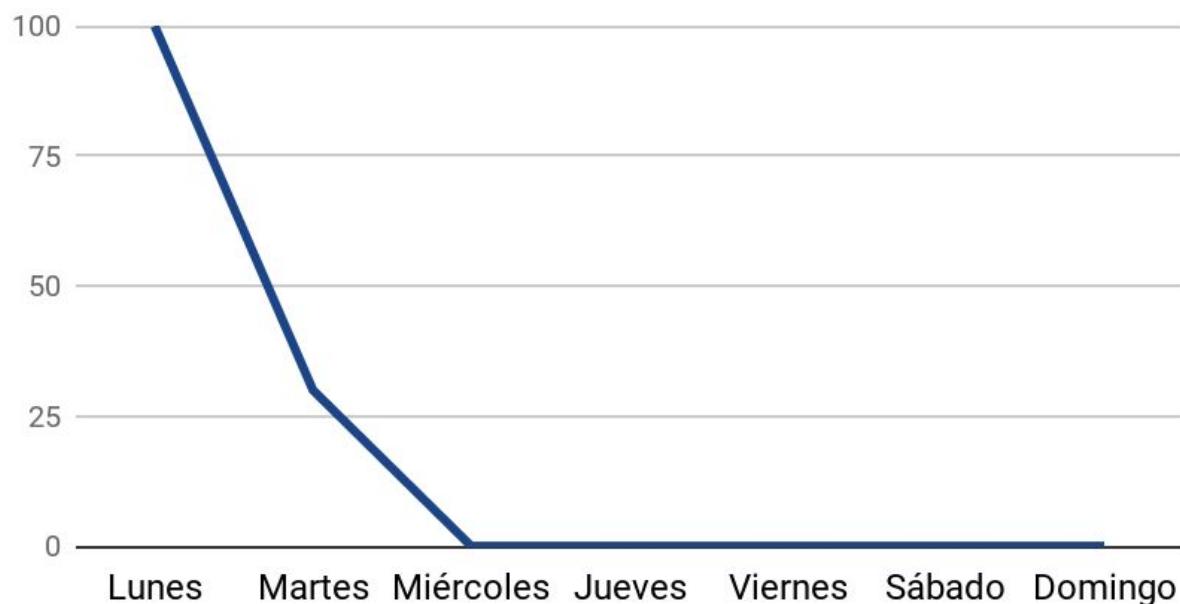
5.3. Sprint 3 (Gestión de Proveedores)

El objetivo de la iteración es tener el CRUD de Proveedores (Suppliers).

5.3.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	20	0	0	0	0	0	0
Vista y Controlador	40	20	0	0	0	0	0
Modelo	20	10	0	0	0	0	0
Test	20	0	0	0	0	0	0
Total	100	30	0	0	0	0	0

Sprint 3



5.3.2. Gemas utilizadas

No hemos utilizado gemas nuevas en este sprint.

5.3.3. Ficheros de migración

El comando que crea las plantillas del modelo, del test del modelo y del fichero de migración de Supplier es el siguiente:

```
$ rails generate model Supplier
```

Modificamos el fichero de migración para generar la tabla:

Fichero: \db\migrate\20190506120000_create_suppliers.rb

```
class CreateSuppliers < ActiveRecord::Migration[5.1]
  def change
    create_table :suppliers do |t|
      t.string :first_name, :limit => 255, :null => false, :unique => true
      t.string :last_name, :limit => 255, :null => false, :unique => true
      t.timestamps
    end
  end
end
```

Para que se creen convenientemente las tablas en la base de datos se creen correctamente debemos realizar la migración con “rake db: migrate”.

5.3.4. Modelos ORM

Dejamos el fichero del modelo de la siguiente forma:

Fichero: app / models / supplier . rb

```
class Supplier < ApplicationRecord
  has_and_belongs_to_many :games
  validates_presence_of :first_name, :message => '(Nombre) no puede dejarse en blanco'
  validates_presence_of :last_name, :message => '(Apellido) no puede dejarse en blanco'
  def name
    "#{first_name} #{last_name}"
  end
end
```

5.3.5. Vistas y controladores

Generamos el controlador supplier, dentro de admin:

```
$ rails generate controller admin/supplier new create edit update destroy show  
index
```

A continuación, modificamos el fichero del controlador:

Fichero: app / controllers / admin / supplier_controller . rb

```
class Admin::SupplierController < ApplicationController  
  def new  
    @supplier = Supplier.new  
    @page_title = 'Crear nuevo proveedor'  
  end  
  
  def create  
    @supplier = Supplier.new(supplier_params)  
    if @supplier.save  
      flash[:notice] = "Proveedor #{@supplier.name} creado correctamente."  
      redirect_to :action => 'index'  
    else  
      @page_title = 'Crear nuevo proveedor'  
      render :action => 'new'  
    end  
  end  
  
  def edit  
    @supplier = Supplier.find(params[:id])  
    @page_title = 'Editar proveedor'  
  end  
  
  def update  
    @supplier = Supplier.find(params[:id])  
    if @supplier.update_attributes(supplier_params)  
      flash[:notice] = "Proveedor #{@supplier.name} actualizado correctamente."  
      redirect_to :action => 'show', :id => @supplier  
    else  
      @page_title = 'Editar proveedor'  
      render :action => 'edit'  
    end  
  end  
  
  def destroy  
    @supplier = Supplier.find(params[:id])  
    @supplier.destroy  
    flash[:notice] = "Proveedor #{@supplier.name} eliminado correctamente."  
    redirect_to :action => 'index'  
  end  
  
  def show  
    @supplier = Supplier.find(params[:id])  
    @page_title = @supplier.name  
  end
```

```

def index
  @suppliers = Supplier.all
  @page_title = 'Listado de proveedores'
end

private
  def supplier_params
    params.require(:supplier).permit(:first_name, :last_name)
  end
end

```

Y seguidamente los ficheros correspondientes a las vistas:

Fichero: app / views / admin / supplier / form.html.erb

```

<% if @supplier.errors.any? %>
  <div id="errorExplanation">
    <h2>Este proveedor no se puede guardar debido a <%= pluralize(@supplier.errors.count, "error", "errores") %></h2>
    <ul>
      <% @supplier.errors.full_messages.each do |msg| %>
        <li><%= msg %></li>
      <% end %>
    </ul>
  </div>
<% end %>

<div class="field">
  <p><label for="supplier_first_name">Nombre</label><br/>
  <%= text_field 'supplier', 'first_name' %></p>
</div>

<div class="field">
  <p><label for="supplier_last_name">Apellidos</label><br/>
  <%= text_field 'supplier', 'last_name' %></p>
</div>

```

Fichero: app / views / admin / supplier / supplier . html . erb

```
<tr>
  <td><%= link_to supplier.name, :action => 'show', :id => supplier %></td>
  <td><%= link_to 'Editar', :action => 'edit', :id => supplier %></td>
  <td>
    <%= button_to 'Eliminar', { :action => 'destroy', :id => supplier },
      data: { confirm: "{Está seguro de que quiere eliminar el proveedor #{supplier.name}}?" } %>
  </td>
</tr>
```

Fichero: app / views / admin / supplier / index . html . erb

```
<table>
  <tr>
    <th>Nombre</th>
    <th>Editar</th>
    <th>Eliminar</th>
  </tr>

  <%= render :partial => 'supplier', :collection => @suppliers %>
</table>

<p><%= link_to 'Añadir un nuevo proveedor', :action => 'new' %></p>
```

Fichero: app / views / admin / supplier / new . html . erb

```
<%= form_tag :action => 'create' do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Crear proveedor' %>
<% end %>

<%= link_to 'Atrás', :action => 'index' %>
```

Fichero: app / views / admin / supplier / show . html . erb

```
<dl>
  <dt>Nombre</dt>
  <dd><%= @supplier.first_name %></dd>
  <dt>Apellidos</dt>
  <dd><%= @supplier.last_name %></dd>
</dl>

<%= link_to 'Editar', :action => 'edit', :id => @supplier %> |
<%= link_to 'Atrás', :action => 'index' %>
```

Fichero: app / views / admin / supplier / edit . html . erb

```
<%= form_tag :action => 'update', :id => @supplier do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Actualizar proveedor' %>
<% end %>

<%= link_to 'Mostrar', :action => 'show', :id => @supplier %>|
<%= link_to 'Atrás', :action => 'index' %>
```

5.3.6. Test

Fichero: test / models / supplier_test . rb

```
require 'test_helper'

class SupplierTest < ActiveSupport::TestCase
  test "test_name" do
    supplier = Supplier.create(:first_name => 'Joel', :last_name => 'Spolsky')
    assert_equal 'Joel Spolsky', supplier.name
  end
end
```

Fichero: test / controllers / admin / supplier_controller_test . rb

```
require 'test_helper'

class Admin::SupplierControllerTest < ActionDispatch::IntegrationTest
  fixtures :suppliers

  test "new" do
    get '/admin/supplier/new'
    assert_template 'admin/supplier/new'
    assert_select 'div#content' do
      assert_select 'h1', 'Crear nuevo proveedor'
      assert_select "form[action=\"/admin/supplier/create\"]"
    end
  end

  test "create" do
    get '/admin/supplier/new'
    assert_template 'admin/supplier/new'
    assert_difference(Supplier, :count) do
      post '/admin/supplier/create', :params => { :supplier => { :first_name => 'Jack', :last_name => 'Sparrow' } }
      assert_response :redirect
      assert_redirected_to :action => 'index'
    end
    assert_equal 'Proveedor Jack Sparrow creado correctamente.', flash[:notice]
  end

  test "failing_create" do
    assert_no_difference(Supplier, :count) do
      post '/admin/supplier/create', :params => { :supplier => { :first_name => 'Jack' } }
      assert_response :success
      assert_template 'admin/supplier/new'
      assert_select "div[class=\"field_with_errors\"]"
    end
  end

  test "edit" do
    get '/admin/supplier/edit', :params => { :id => 1 }
    assert_select 'input' do
      assert_select '[type=?]', 'text'
      assert_select '[name=?]', 'supplier[first_name]'
      assert_select '[value=?]', 'Joel'
    end
    assert_select 'input' do
      assert_select '[type=?]', 'text'
      assert_select '[name=?]', 'supplier[last_name]'
      assert_select '[value=?]', 'Spolsky'
    end
  end

  test "update" do
    post '/admin/supplier/update', :params => { :id => 1, :supplier => { :first_name => 'Joseph', :last_name => 'Spolsky' } }
    assert_response :redirect
    assert_redirected_to :action => 'show', :id => 1
    assert_equal 'Joseph', Supplier.find(1).first_name
  end

  test "test_destroy" do
    assert_difference(Supplier, :count, -1) do
      post '/admin/supplier/destroy', :params => { :id => 1 }
    end
  end
end
```

```

    assert_equal flash[:notice], 'Proveedor Joel Spolsky eliminado correctamente.'
    assert_response :redirect
    assert_redirected_to :action => 'index'
    get '/admin/supplier/index'
    assert_response :success
    assert_select 'div#notice', 'Proveedor Joel Spolsky eliminado correctamente.'
end
end

test "show" do
  get '/admin/supplier/show', :params => { :id => 1 }
  assert_template 'admin/supplier/show'
  assert_equal 'Joel', assigns(:supplier).first_name
  assert_equal 'Spolsky', assigns(:supplier).last_name
  assert_select 'div#content' do
    assert_select 'h1', Supplier.find(1).name
  end
end

test "index" do
  get '/admin/supplier/index'
  assert_response :success
  assert_select 'table' do
    assert_select 'tr', Supplier.count + 1
  end
  Supplier.find_each do |a|
    assert_select 'td', a.name
  end
end
end

```

5.3.7. Dificultades encontradas

El proceso para gestionar el proveedor es casi idéntico al de fabricante, por lo tanto no hemos encontrado grandes dificultades.

5.3.8. Objetivos alcanzados

Visualizar el CRUD (Create, Read, Update and Delete) de Proveedor.

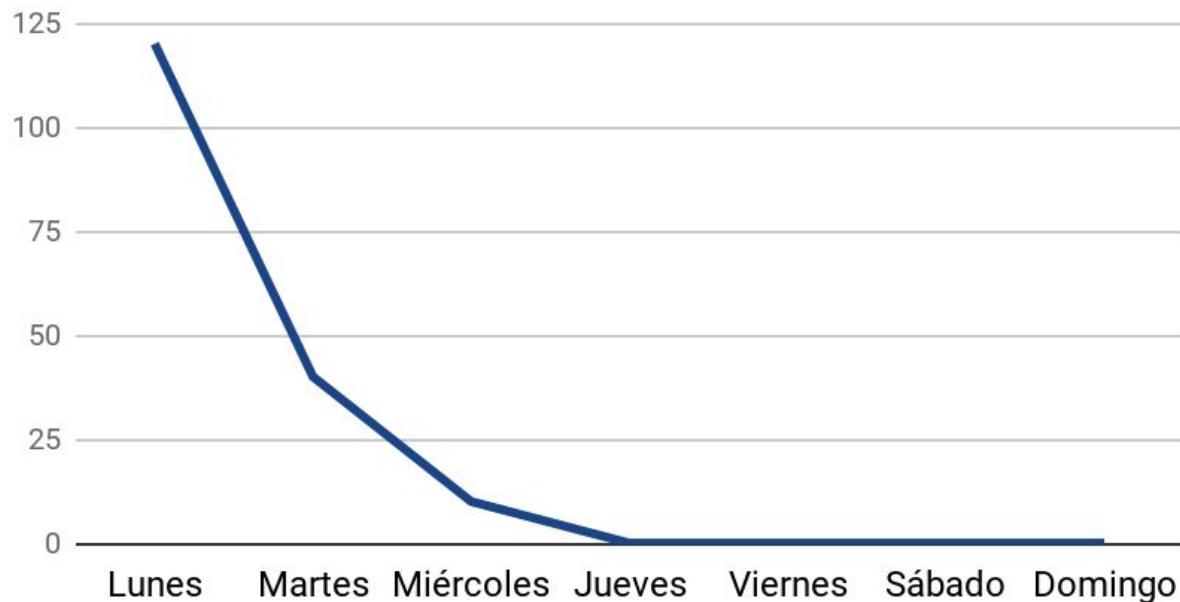
5.4. Sprint 4 (Gestión de Artículos)

El objetivo de esta iteración es tener el CRUD de Artículos, nuestra entidad principal y su asociación con la entidad secundaria Manufacturer, cuya relación es de uno a muchos. Y también con la asociación con la entidad secundaria Supplier, cuya relación es de muchos a muchos. Para este Sprint hemos contado con dos semanas debido a las fiestas de Semana Santa. En esta ocasión, además del test del modelo, tuvimos que realizar los correspondientes tests de integración.

5.4.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	20	0	0	0	0	0	0
Vista y Controlador	50	30	10	0	0	0	0
Modelo	30	10	0	0	0	0	0
Test	20	0	0	0	0	0	0
Total	120	40	10	0	0	0	0

Sprint 4



5.4.2. Gemas utilizadas

En esta iteración si hemos incluido nuevas gemas:

- will_paginate, versión 3.0.7: Paginación.
- RedCloth, versión 4.2.9 : Añade estilo al texto plano para html en la descripción de los artículos
- paperclip, versión 4.3.2 : Adjunta imágenes.

5.4.3. Ficheros de migración

Creamos el fichero de migración y el fichero de modelo de Article (Game), sin que cree el fichero de migración, porque ya lo creamos anteriormente (--skip-migration).

```
ine@gii ~/projects/eshop1 $ rails generate migration  
create_games_and_game_suppliers
```

Modificamos el fichero de migración para generar la tabla:

Fichero: db\migrate\20190506120002_create_games_and_games_suppliers.rb

```
class CreateGamesAndGamesSuppliers < ActiveRecord::Migration[5.1]  
  def up  
    create_table :games do |t|  
      t.string :name, :limit => 255, :null => false  
      t.integer :developer_id, :limit => 8, :null => false  
      t.datetime :developed_at  
      t.text :blurb  
      t.integer :pegi  
      t.float :price  
      t.timestamps  
    end  
  
    create_table :games_suppliers do |t|  
      t.integer :game_id, :limit => 8, :null => false  
      t.integer :supplier_id, :limit => 8, :null => false  
      t.timestamps  
    end  
  
    say_with_time 'Adding foreing keys' do  
      # Add foreign key reference to games_suppliers table  
      execute 'ALTER TABLE games_suppliers ADD CONSTRAINT fk_games_suppliers_suppliers  
              FOREIGN KEY (supplier_id) REFERENCES suppliers(id) ON DELETE CASCADE'  
      execute 'ALTER TABLE games_suppliers ADD CONSTRAINT fk_games_suppliers_games  
              FOREIGN KEY (game_id) REFERENCES games(id) ON DELETE CASCADE'  
      # Add foreign key reference to developers table  
      execute 'ALTER TABLE games ADD CONSTRAINT fk_games_developers  
              FOREIGN KEY (developer_id) REFERENCES developers(id) ON DELETE CASCADE'  
    end  
  end  
  
  def self.down  
    drop_table :games  
    drop_table :games_suppliers  
  end  
end
```

Para la inserción de imágenes de artículos añadimos el siguiente fichero dentro de db/migrate:

Fichero: db\migrate\20190506120003_add_cover_image_attachment_to_games.rb

```
class AddCoverImageAttachmentToGames < ActiveRecord::Migration[5.1]
  def up
    add_attachment :games, :cover_image
  end

  def down
    remove_attachment :games, :cover_image
  end
end
```

Creamos ahora el fichero del modelo:

```
ine@gii ~/projects/eshop1 $ rails generate model Game --skip-migration
```

Y editamos dicho fichero:

Fichero: app / models / game. rb

```
class Game < ApplicationRecord

  has_and_belongs_to_many :suppliers
  belongs_to :developer

  #has_many :cart_items
  #has_many :carts, :through => :cart_items

  has_attached_file :cover_image
  validates_attachment :cover_image,
  :content_type => { :content_type => ["image/jpeg", "image/gif", "image/png"] }

  validates_length_of :name, :in => 1..255, :message => '(Nombre) es muy corto, debe contener al menos una letra'
  validates_presence_of :developer
  validates_presence_of :suppliers, :message => '(Proveedor) no seleccionado'
  validates_presence_of :developed_at
  validates_length_of :pegi, :in => 1..2, :message => 'debe contener al menos dos cifras'
  validates_numericality_of :price, :message => '(Precio) inexistente'

  def supplier_names
    self.suppliers.map{|supplier| supplier.name}.join(", ")
  end

  #def self.latest(num)
  #  all.order("games.id desc").includes(:suppliers, :developer).limit(num)
  #end
end
```

Finalmente realizamos la migración con:

```
ine@jii ~/projects/eshop1 $ rake db:migrate
```

5.4.4. Vistas y controladores

Generamos el controlador supplier, dentro de admin:

```
ine@jii ~/projects/eshop1 $ rails generate controller admin/article new create edit  
update destroy show index
```

A continuación, modificamos el fichero del controlador:

Fichero: app / controllers / admin / game_controller . rb

```
class Admin::GameController < ApplicationController  
  def new  
    load_data  
    @game = Game.new  
    @page_title = 'Crear nuevo juego'  
  end  
  
  def create  
    @game = Game.new(game_params)  
    if @game.save  
      flash[:notice] = "Juego #{@game.name} creado correctamente."  
      redirect_to :action => 'index'  
    else  
      load_data  
      @page_title = 'Crear nuevo juego'  
      render :action => 'new'  
    end  
  end  
  
  def edit  
    load_data  
    @game = Game.find(params[:id])  
    @page_title = 'Editar juego'  
  end  
  
  def update  
    @game = Game.find(params[:id])  
    if @game.update_attributes(game_params)  
      flash[:notice] = "Juego #{@game.name} actualizado correctamente."  
      redirect_to :action => 'show', :id => @game  
    else  
      load_data  
      @page_title = 'Editar juego'  
      render :action => 'edit'  
    end  
  end  
  
  def destroy  
    @game = Game.find(params[:id])  
    @game.destroy  
    flash[:notice] = "Juego #{@game.name} eliminado correctamente."  
    redirect_to :action => 'index'  
  end
```

```

def show
  @game = Game.find(params[:id])
  @page_title = @game.name
end

def index
  sort_by = params[:sort_by]
  @games = Game.order(sort_by).paginate(:page => params[:page], :per_page => 5)
  @page_title = 'Listado de juegos'
end

private

def load_data
  @suppliers = Supplier.all
  @developers = Developer.all
end

def game_params
  params.require(:game).permit(:name, :developer_id, :developed_at, { :supplier_ids => [] },
                               :blurb, :pegi, :price, :cover_image)
end
end

```

Y después los ficheros correspondientes a las vistas

Fichero: app / views / admin / game / form . html . erb

```

<% if @game.errors.any? %>
<div id="errorExplanation">
<h2>Este juego no se puede guardar debido a <%= pluralize(@game.errors.count, "error", "errores") %></h2>
<ul>
<% @game.errors.full_messages.each do |msg| %>
<li><%= msg %></li>
<% end %>
</ul>
</div>
<% end %>

<div class="field">
<p><label for="game_name">Nombre</label><br/>
<%= text_field 'game', 'name' %></p>
</div>

<div class="field">
<p><label for="game_developer">Desarrollador</label><br/>
<%= collection_select :game, :developer_id, @developers, :id, :name %></p>
</div>

<div class="field">
<p><label for="game[supplier_ids][]">Proveedores</label><br/>
<%= select_tag 'game[supplier_ids][]"', options_from_collection_for_select(@suppliers, :id, :name,
@game.suppliers.collect{|supplier| supplier.id}), { :multiple => true, :size => 5 } %></p>
</div>

```

```

<div class="field"><p>
  <label for="game_developed_at">Desarrollado el</label><br/>
  <%= datetime_select 'game', 'developed_at' %></p>
</div>

<div class="field">
  <p><label for="game_blurb">Descripción</label><br/>
  <%= text_area 'game', 'blurb' %></p>
</div>

<div class="field">
  <p><label for="game_pegi">PEGI</label><br/>
  <%= text_field 'game', 'pegi' %></p>
</div>

<div class="field">
  <p><label for="game_price">Precio</label><br/>
  <%= text_field 'game', 'price' %></p>
</div>

<div class="field">
  <% if @game.cover_image.exists? then %>
    <dd><%= image_tag @game.cover_image.url %></dd>
  <% else %>
    <p> No hay imagen de portada. Por favor, adjunte una. </p>
  <% end %>
  <p><label for="game_cover_image">Imagen de portada</label><br/>
  <%= file_field 'game', :cover_image %></p>
</div>

```

Fichero: app / views / admin / article / edit . html . erb

```

<%= form_tag "/admin/game/update?id=#{@game.id}", :multipart => true do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Actualizar juego' %>
<% end %>

<%= link_to 'Mostrar', :action => 'show', :id => @game %> |
<%= link_to 'Atrás', :action => 'index' %>

```

Fichero: app / views / admin / article / index . html . erb

```
<table>
  <tr>
    <th><a href=?sort_by=developer_id?>Desarrollador</a></th>
    <th><a href=?sort_by=name?>Nombre</a></th>
    <th><a href=?sort_by=pegi?>PEGI</a></th>
    <th colspan="3"></th>
  </tr>

  <% @games.each do |game| %>
  <tr>
    <td><%= h game.developer.name %></td>
    <td><%= h game.name %></td>
    <td><%= h game.pegi %></td>
    <td><%= link_to 'Mostrar', :action => 'show', :id => game %></td>
    <td><%= link_to 'Editar', :action => 'edit', :id => game %></td>
    <td><%= button_to 'Eliminar', { :action => 'destroy', :id => game },>
        data: { confirm: "¿Está seguro de que quiere eliminar el juego #{game.name}?" } %>
    </td>
  </tr>
  <% end %>
</table>
<br/>
<%= will_paginate @games, :page_links => false, :link_separator => ' | ',>
      :previous_label => 'Página anterior', :next_label => 'Página siguiente' %>
<p><%= link_to 'Añadir un nuevo juego', :action => 'new' %></p>
```

Fichero: app / views / admin / article / new . html . erb

```
<%= form_tag "/admin/game/create", :multipart => true do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Crear Juego' %>
<% end %>

<%= link_to 'Atrás', :action => 'index' %>
```

Fichero: app / views / admin / article / show . html . erb

```

<dl>
  <dt>Nombre</dt>
  <dd><%= @game.name %></dd>
  <dt>Desarrollador</dt>
  <dd><%= @game.developer.name %></dd>
  <dt>Desarrollado el el</dt>
  <dd><%= @game.developed_at.strftime("%d/%m/%Y a las %I:%M%p") %></dd>
  <dt>Proveedor</dt>
  <dd><%= @game.supplier_names %></dd>
  <dt>Descripción</dt>
  <%= RedCloth.new(@game.blurb).to_html.html_safe if @game.blurb %>
  <dt>PEGI</dt>
  <dd><%= @game.pegi %></dd>
  <dt>Precio</dt>
  <dd><%= @game.price %></dd>
  <dt>Imagen de portada</dt>
  <% if @game.cover_image.exists? then %>
    <dd><%= image_tag @game.cover_image.url %></dd>
  <% else %>
    <dd><%= image_tag '/missing.png' %></dd>
    <p> No hay imagen de portada. Por favor, adjunte una. </p>
  <% end %>
</dl>

<%= link_to 'Editar', :action => 'edit', :id => @game %> |
<%= link_to 'Atrás', :action => 'index' %>

```

5.4.5. Test

Preparamos el fichero de los tests, primero el modelo:

Fichero: test / models / game_test . rb

```

require 'test_helper'

class GameTest < ActiveSupport::TestCase
  fixtures :suppliers, :developers, :games, :games_suppliers

  test "failing_create" do
    game = Game.new
    assert_equal false, game.save
    assert_equal 7, game.errors.count
    assert game.errors[:name]
    assert game.errors[:developer]
  end
end

```

```

    assert game.errors[:suppliers]
    assert game.errors[:developed_at]
    assert game.errors[:blurb]
    assert game.errors[:pegi]
    assert game.errors[:price]
end

test "create" do
  game = Game.new(
    :name => 'Ruby on Rails',
    :suppliers => Supplier.all,
    :developer_id => Developer.find(1).id,
    :developed_at => Time.now,
    :blurb => 'A great game',
    :pegi => 7,
    :price => 45.5
  )
  assert game.save
end

test "has_many_and_belongs_to_mapping" do
  apress = Developer.find_by_name("Apress");
  count = apress.games.count
  game = Game.new(
    :name => 'Pro Rails E-Commerce 8th Edition',
    :suppliers => [Supplier.find_by_first_name_and_last_name('Joel', 'Spolsky'),
      Supplier.find_by_first_name_and_last_name('Jeremy', 'Keith')],
    :developer_id => apress.id,
    :developed_at => Time.now,
    :blurb => 'E-Commerce on Rails',
    :pegi => 18,
    :price => 55.5
  )
  apress.games << game
  apress.reload
  game.reload
  assert_equal count + 1, apress.games.count
  assert_equal 'Apress', game.developer.name
end

test "has_many_and_belongs_to_many_suppliers_mapping" do
  game = Game.new(
    :name => 'Pro Rails E-Commerce 8th Edition',
    :suppliers => [Supplier.find_by_first_name_and_last_name('Joel', 'Spolsky'),
      Supplier.find_by_first_name_and_last_name('Jeremy', 'Keith')],
    :developer_id => Developer.find_by_name("Apress").id,
    :developed_at => Time.now,
    :blurb => 'E-Commerce on Rails',
    :pegi => 12,
    :price => 55.5
  )
  assert game.save
  game.reload
  assert_equal 2, game.suppliers.count
  assert_equal 2, Supplier.find_by_first_name_and_last_name('Joel', 'Spolsky').games.count
end
end

```

Crearemos el fichero para el test de Integración de la entidad principal y modificamos el fichero correspondiente:

Fichero: test / integration / game_administration_test . rb

```
require 'test_helper'

class GameAdministrationTest < ActionDispatch::IntegrationTest

  test "game_administration" do
    developer = Developer.create(:name => 'Games of Ruby')
    supplier = Supplier.create(:first_name => 'John', :last_name => 'Anderson')
    george = new_session_as(:george)

    new_game_ruby = george.add_game :game => {
      :name => 'A new Game of Ruby',
      :developer_id => developer.id,
      :supplier_ids => [supplier.id],
      :developed_at => Time.now,
      :blurb => 'A new Game of Ruby',
      :pegi => 3,
      :price => 45.5
    }

    george.list_games
    george.show_game new_game_ruby

    george.edit_game new_game_ruby, :game => {
      :name => 'A very new Game of Ruby',
      :developer_id => developer.id,
      :supplier_ids => [supplier.id],
      :developed_at => Time.now,
      :blurb => 'A very new Game of Ruby',
      :pegi => 18,
      :price => 50
    }

    bob = new_session_as(:bob)
    bob.delete_game new_game_ruby
  end

  private

  module GameTestDSL
    attr_writer :name

    def add_game(parameters)
      supplier = Supplier.first
      developer = Developer.first
      get '/admin/game/new'
      assert_response :success
      assert_template 'admin/game/new'
      assert_select 'select#game_developer_id' do
        assert_select "option[value=\"#{developer.id}\"]", developer.name
      end
      assert_select "select[name=\"game[supplier_ids][]\"]" do
        assert_select "option[value=\"#{supplier.id}\"]", supplier.name
      end
    end
  end
end
```

```

post '/admin/game/create', :params => parameters
assert_response :redirect
follow_redirect!
assert_response :success
assert_template 'admin/game/index'
page = Game.all.count / 5 + 1
get "/admin/game/index?page=#{page}"
assert_select 'td', parameters[:game][:name]
game = Game.find_by_name(parameters[:game][:name])
return game;
end

def edit_game(game, parameters)
  get "/admin/game/edit?id=#{game.id}"
  assert_response :success
  assert_template 'admin/game/edit'
  post "/admin/game/update?id=#{game.id}", :params => parameters
  assert_response :redirect
  follow_redirect!
  assert_response :success
  assert_template 'admin/game/show'
end

def delete_game(game)
  post "/admin/game/destroy?id=#{game.id}"
  assert_response :redirect
  follow_redirect!
  assert_template 'admin/game/index'
end

def show_game(game)
  get "/admin/game/show?id=#{game.id}" #cambiado de book_administration_test
  assert_response :success
  assert_template 'admin/game/show'
end

def list_games
  get '/admin/game/index'
  assert_response :success
  assert_template 'admin/game/index'
end
end

def new_session_as(name)
  open_session do |session|
    session.extend(GameTestDSL)
    session.name = name
    yield session if block_given?
  end
end
end

```

Vemos que todo va bien ejecutando el test:

```
ine@giil ~/projects/eshop1 $ rake test TEST=test/models/game_test.rb
```

5.4.6. Dificultades encontradas

En este sprint la principal dificultad la hemos tenido en establecer la relación entre la entidad principal y las secundarias. Del mismo modo el test de integración ha sido novedoso para nosotros y tuvimos algún problema para entender su funcionamiento al principio.

5.4.7. Objetivos alcanzados

Con este sprint hemos conseguido poder visualizar el CRUD (Create, Read, Update and Delete) de Game y su relación con las entidades secundarias. Asimismo se ha añadido la inserción de imágenes y la paginación del listado.

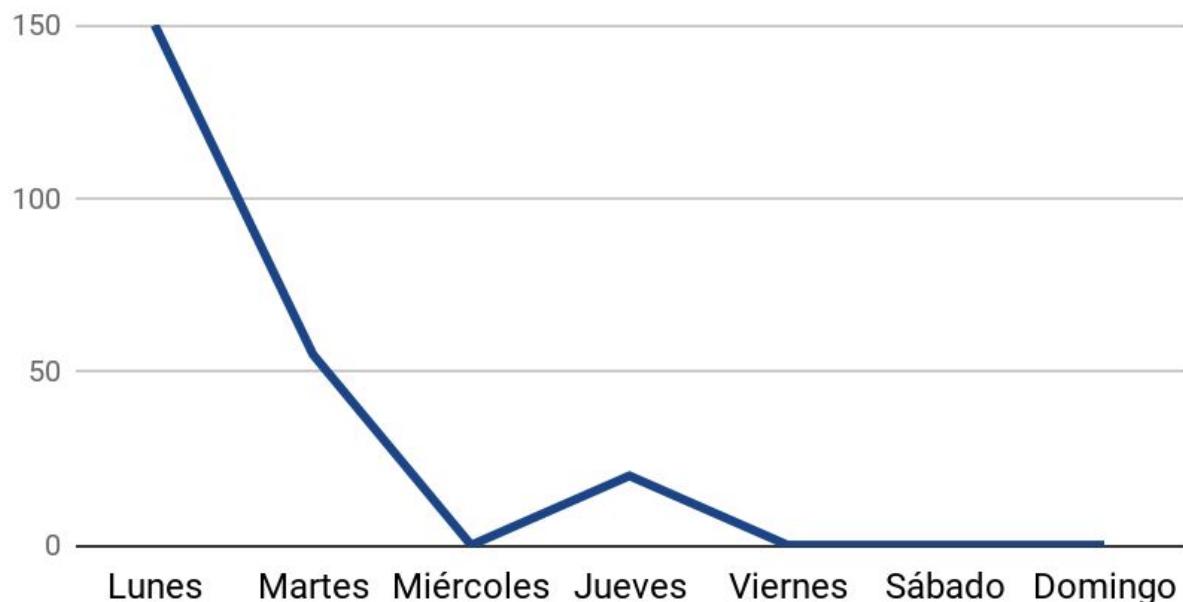
5.5. Sprint 5 (Creación del catálogo)

Ahora que ya tenemos los CRUD necesarios para nuestra tienda, lo que nos hace falta es tener un catálogo operativo para que los clientes puedan ver nuestros artículos a partir de la información almacenada gracias a lo creado anteriormente. Vamos a crear el catálogo junto a su test correspondiente.

5.5.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	0	0	0	0	0	0	0
Vista y Controlador	90	35	0	10	0	0	0
Modelo	40	20	0	0	0	0	0
Test	20	0	0	10	0	0	0
Total	150	55	0	20	0	0	0

Sprint 5



5.5.2. Gemas utilizadas

En este sprint no hemos añadido gemas nuevas.

5.5.3. Ficheros de migración

En este sprint no hemos añadido ficheros de migración.

5.5.4. Modelos ORM

En este sprint no ha sido necesario crear ningún modelo, ya que la información de la base de datos que vamos a mostrar ya está gestionada en otros modelos (game, manufacturer y supplier)

5.5.5. Vistas y controladores

Generamos el controlador y lo modificamos:

```
ine@gii ~/projects/eshop1 $ rails generate controller Catalog index show latest
```

El fichero debe quedar así:

Fichero: app / controllers / catalog_controller . rb

```
class CatalogController < ApplicationController
# before_action :initialize_cart, :except => :show
# before_action :require_no_user

  def show
    @game = Game.find(params[:id])
    @page_title = @game.name
  end

  def index
    @games = Game.order("games.id desc").includes(:suppliers, :developer).paginate(:page => params[:page], :per_page => 5)
    @page_title = 'Catálogo'
  end

  def latest
    @games = Game.latest 5 # invoques "latest" method to get the five latest games
    @page_title = 'Últimos juegos'
  end
end
```

Hay que acordarse de descomentar las últimas líneas del Fichero: app/models/article.rb (apartado 5.4.3) , donde está definido el método latest:

Fichero: app / models / game . rb (añadir o descomentar el siguiente método)

```
def self.latest(num)
  all.order("games.id desc").includes(:suppliers, :developer).limit(num)
end
```

Y ahora las vistas:

Fichero: app / views / catalog / games . html . erb

```
<dl id = 'games'>
  <% for game in @games %>
    <dt>
      <%= link_to game.name, :action => 'show', :id => game %>
      <%= link_to '+', :controller => 'cart', :action => 'add', :id => game %>
    </dt>
    <dd><%= game.developer.name %></dd>
    <dd><%= sprintf("Precio: %.2f €", game.price) %> </dd>
    <dd><small>Proveedores:</small>
      <% for supplier in game.suppliers %>
        <%= supplier.last_name %>, <%= supplier.first_name %>.
      <% end %>
    </small></dd>
  <% end %>
</dl>
```

Fichero: app / views / catalog / index . html .

```
<%= render :partial => 'games' %>
<%= will_paginate @games, :page_links => false, :link_separator => ' | ',
  :previous_label => 'Página anterior', :next_label => 'Página siguiente' %>
```

Fichero: app / views / catalog / latest . html . erb

```
<%= render :partial => 'games' %>
```

Fichero: app / views / catalog / show . html . erb

```
<h2>por <%= @game.developer.name %></h2>
<% if @game.cover_image.exists? then %>
  <dd><%= image_tag @game.cover_image.url %></dd>
<% else %>
  <p>Imagen de portada no disponible.</p>
<% end %>
<dl>
  <dt>Precio</dt>

  <dd><%= sprintf("%.2f €", @game.price) %> </dd>
  <dt>PEGI</dt>
  <dd><%= @game.pegi %></dd>
  <dt>Proveedores</dt>
  <dd><%= @game.supplier_names %></dd>
  <dt>Descripción</dt>
  <%= RedCloth.new(@game.blurb).to_html.html_safe if @game.blurb %>
</dl>

<p><%= link_to 'Catálogo', :action => 'index' %> </p>
```

5.5.6. Test

Vamos a realizar un test de integración, primero crearemos el fichero correspondiente con:

```
ine@gi ~ / projects / eshop1 $ rake test
TEST=test/integration/browsing_and_searching_test.rb
```

El fichero queda de la siguiente forma:

Fichero: test / integration / browsing_and_searching_test . rb

```
require 'test_helper'

class BrowsingTest < ActionDispatch::IntegrationTest
  fixtures :developers, :suppliers, :games, :games_suppliers

  test "browse" do
    jill = new_session_as :jill
    jill.index
    jill.second_page
    jill.game_details 'Pride and Prejudice'
    jill.latest_games
  end

  module BrowsingTestDSL
    include ERB::Util
    attr_writer :name

    def index
      get '/catalog/index'
      assert_response :success
      assert_select 'dl#games' do
        assert_select 'dt', :count => 5
      end
      assert_select 'dt' do
        assert_select 'a', 'The Idiot'
      end
      check_game_links
    end

    def second_page
      get '/catalog/index?page=2'
      assert_response :success
      assert_template 'catalog/index'
      assert_equal Game.find_by_name('Pro Rails E-Commerce'),
                   assigns(:games).last
      check_game_links
    end
  end
end
```

```

def game_details(name)
  @game = Game.where(:name => name).first
  get "/catalog/show/#{@game.id}"
  assert_response :success
  assert_template 'catalog/show'
  assert_select 'div#content' do
    assert_select 'h1', @game.name
    assert_select 'h2', "por #{@game.developer.name}"
  end
end

def latest_games
  get '/catalog/latest'
  assert_response :success
  assert_template 'catalog/latest'
  assert_select 'dl#games' do
    assert_select 'dt', :count => 5
  end
  @games = Game.latest(5)
  @games.each do |a|
    assert_select 'dt' do
      assert_select 'a', a.name
    end
  end
end

def check_game_links
  for game in assigns :games
    assert_select 'a' do
      assert_select '[href=?]', "/catalog/show/#{game.id}"
    end
  end
end

def new_session_as(name)
  open_session do |session|
    session.extend(BrowsingTestDSL)
    session.name = name
    yield session if block_given?
  end
end

```

Después ejecutamos el test con:

```
ine@gii ~/projects/eshop1 $ rake test  
TEST=test/integration/browsing_and_searching_test.rb
```

5.5.7. Dificultades encontradas

En este sprint no hemos encontrado dificultades.

5.5.8. Objetivos alcanzados

Con este sprint hemos conseguido crear el catálogo de nuestros artículos.

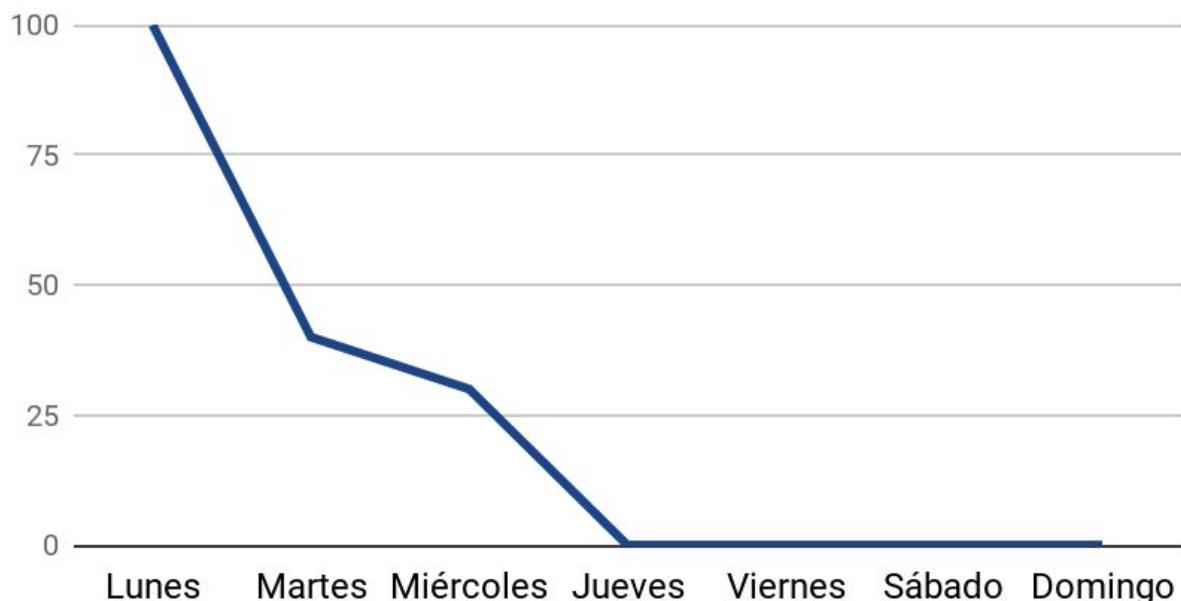
5.6. Sprint 6 (Carrito de la compra)

Una vez creado el catálogo es necesario añadir una funcionalidad para que podamos seleccionar los artículos mostrados para poder comprarlos. Podremos añadir y eliminar artículos al carrito, así como vaciar de una sola vez dicho carrito. Para ello vamos a necesitar integrar en nuestro proyecto jQuery

5.6.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	10	10	0	0	0	0	0
Vista y Controlador	50	20	20	0	0	0	0
Modelo	20	10	10	0	0	0	0
Test	20	0	0	0	0	0	0
Total	100	40	30	0	0	0	0

Sprint 6



5.6.2. Gemas utilizadas

En este sprint hemos necesitado añadir la gema:

- jquery-rails, versión 4.0.5

5.6.3. Ficheros de migración

Creamos las plantillas del modelo, del test del modelo y del fichero de migración del carrito de la compra:

```
ine@gii ~/projects/eshop1 $ rails generate model Cart
ine@gii ~/projects/eshop1 $ rails generate model CartItem
```

Editamos pues los ficheros de migración:

Fichero: db\migrate\20190506120004_create_carts.rb

```
class CreateCarts < ActiveRecord::Migration[5.1]
  def change
    create_table :carts do |t|
      t.timestamps
    end
  end
end
```

Fichero: db\migrate\20190506120005_create_cart_items.rb

```
class CreateCartItems < ActiveRecord::Migration[5.1]
  def change
    create_table :cart_items do |t|
      t.integer :game_id, :limit => 8
      t.integer :cart_id, :limit => 8
      t.float :price
      t.integer :amount
      t.timestamps
    end
  end
end
```

Para que se creen las tablas en la base de datos debemos realizar la migración:

```
ine@gii ~/projects/eshop1 $ rake db:migrate
```

5.6.4. Modelos ORM

Los ficheros correspondiente al modelo quedarían de la siguiente forma:

Fichero: app / models / cart . rb

```
class Cart < ApplicationRecord
  has_many :cart_items
  has_many :games, :through => :cart_items

  def add(game_id)
    items = cart_items.where(game_id: game_id)
    game = Game.find game_id
    if items.size < 1
      ci = cart_items.create :game_id => game_id, :amount => 1, :price => game.price
    else
      ci = items.first
      ci.update_attribute :amount, ci.amount + 1
    end
    ci
  end

  def remove(game_id)
    ci = cart_items.where(game_id: game_id).first
    if ci.amount > 1
      ci.update_attribute :amount, ci.amount - 1
    else
      CartItem.destroy ci.id
    end
    ci
  end

  def total
    sum = 0
    cart_items.each do |item| sum += item.price * item.amount end
    sum
  end
end
```

Fichero: app / models / cart_item . rb

```
class CartItem < ApplicationRecord
  belongs_to :cart
  belongs_to :game
end
```

5.6.5. Vistas y controladores

Generamos el controlador:

```
ine@gii ~/projects/eshop1 $ rails generate controller Cart
```

Editamos el fichero del controlador del carrito:

Fichero: app / controllers / cart_controller . rb

```
class CartController < ApplicationController
  before_action :initialize_cart

  def add
    @game = Game.find params[:id]
    @page_title = 'Añadir artículo'
    if request.post?
      @item = @cart.add params[:id]
      flash[:cart_notice] = "Añadido <em>#{@item.game.name}</em>."
      redirect_to :controller => 'catalog'
    else
      render :controller => 'cart', :action => 'add', :template => 'cart/add'
    end
  end

  def remove
    @game = Game.find params[:id]
    @page_title = 'Eliminar artículo'
    if request.post?
      @item = @cart.remove params[:id]
      flash[:cart_notice] = "Eliminado <em>#{@item.game.name}</em>."
      redirect_to :controller => 'catalog'
    else
      render :controller => 'cart', :action => 'remove', :template => 'cart/remove'
    end
  end

  def clear
    @page_title = 'Vaciar carrito'
    if request.post?
      @cart.cart_items.destroy_all
      flash[:cart_notice] = "Carrito vaciado."
      redirect_to :controller => 'catalog'
    else
      render :controller => 'cart', :action => 'clear', :template => 'cart/clear'
    end
  end
end
```

Seguidamente vamos a mostrar los ficheros correspondientes a las vistas:

Fichero: app / views / cart / cart . html . erb

```
<% if flash[:cart_notice] %>
<%= render :partial => 'cart/cart_notice' %>
<% end %>

<h3>Su carrito de la compra</h3>
<ul>
  <% for item in @cart.cart_items %>
    <li id="cart_item_<%= item.game.id %>">
      <%= render :partial => 'cart/item', :object => item %>
    </li>
  <% end %>
</ul>
<p id='cart_total'><strong>Total: <%= sprintf "%0.2f €", @cart.total %></strong></p>
<% unless @cart.cart_items.empty? %>
  <p id='clear_cart_link'>
    <b><%= link_to 'Vaciar carrito', :controller => 'cart', :action => 'clear' %></b>
  </p>
<% end %>
```

Fichero: app / views / cart / cart_notice . html . erb

```
<p id='cart_notice'><%= flash[:cart_notice].html_safe if flash[:cart_notice] %></p>
```

Fichero: app / views / cart / item . html . erb

```
<%= link_to item.game.name, :action => 'show', :controller => 'catalog', :id => item.game.id %>
<%= pluralize item.amount, "juego", "juegos" %>, <%= sprintf "%0.2f €", item.price * item.amount %>
(<%= link_to '<b>-</b>'.html_safe, :controller => 'cart', :action => 'remove', :id => item.game %>)
```

Fichero: app / views / cart / add . html . erb

```
<strong>Por favor, confirme la agregación de <em><%= @game.name %></em> al carrito de la compra.</strong>
<br><br>
<div style="float: left; width: auto;">
<%= button_to 'Cancelar', { :controller => 'catalog', :action => 'index' }, :method => :get %>
</div>
<div style="float: left; width: auto;">
<%= button_to 'Confirmar', :action => 'add', :id => params[:id] %>
</div>
<br>
```

Fichero: app / views / cart / clear . html . erb

```
<strong>Por favor, confirme el vaciado del carrito de la compra.</strong>
<br><br>
<div style="float: left; width: auto;">
<%= button_to 'Cancelar', { :controller => 'catalog', :action => 'index' }, :method => :get %>
</div>
<div style="float: left; width: auto;">
<%= button_to 'Confirmar', :action => 'clear', :id => params[:id] %>
</div>
<br>
```

Fichero: app / views / cart / remove . html . erb

```
<strong>Por favor, confirme la eliminación de <em><%= @game.name %></em> del carrito de la compra.</strong>
<br><br>
<div style="float: left; width: auto;">
<%= button_to 'Cancelar', { :controller => 'catalog', :action => 'index' }, :method => :get %>
</div>
<div style="float: left; width: auto;">
<%= button_to 'Confirmar', :action => 'remove', :id => params[:id] %>
</div>
<br>
```

Hay que acordarse de descomentar 2 líneas que indicamos del Fichero:
app/models/game.rb:

```
has_many :cart_items
has_many :carts, :through => :cart_items
```

Tenemos que descomentar la siguiente línea del fichero app/controllers/catalog_controller.rb

Fichero: app / controllers / catalog_controller.rb (añadimos o descomentamos esta línea del código)

```
before_action :initialize_cart, :except => :show
```

Ejecutamos el test:

```
ine@gii ~/projects/eshop1 $ rake test TEST=test/controllers/cart_controller_test.rb
```

5.6.6. Test

Ahora pasamos a modificar el controlador del carrito de la compra:

Fichero: test / controllers / cart_controller_test . rb

```
require 'test_helper'

class CartControllerTest < ActionDispatch::IntegrationTest
  fixtures :suppliers, :developers, :games

  test "add" do
    assert_difference(CartItem, :count) do
      post '/cart/add', :params => { :id => 4 }
    end
    assert_response :redirect
    assert_redirected_to :controller => 'catalog'
    assert_equal 1, Cart.find(@request.session[:cart_id]).cart_items.size
  end

  test "remove" do
    post '/cart/add', :params => { :id => 4 }
    assert_equal [Game.find(4)], Cart.find(@request.session[:cart_id]).games

    post '/cart/remove', :params => { :id => 4 }
    assert_equal [], Cart.find(@request.session[:cart_id]).games
  end

  test "clear" do
    post '/cart/add', :params => { :id => 4 }
    assert_equal [Game.find(4)], Cart.find(@request.session[:cart_id]).games

    post '/cart/clear'
    assert_response :redirect
    assert_redirected_to :controller => 'catalog'
    assert_equal [], Cart.find(@request.session[:cart_id]).games
  end
end
```

Para ejecutar el test:

```
ine@gii ~/projects/eshop1 $ rake test  
TEST=test/controllers/cart_controller_test.rb
```

5.6.7. Dificultades encontradas

En este sprint no hemos encontrado dificultades importantes.

5.6.8. Objetivos alcanzados

Hemos conseguido implementar el carrito de la compra que recoge los artículos y su precio y se puede añadir, eliminar y borrar todo el carrito.

5.7. Sprint 7 (Facturación y pedidos)

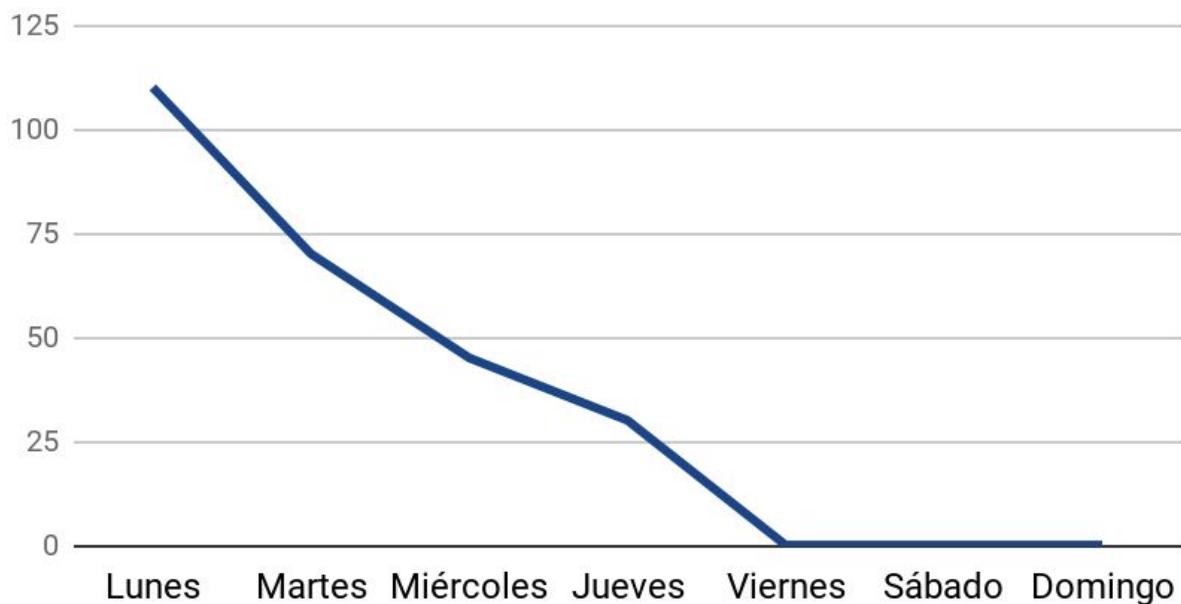
En esta iteración hemos desarrollado dos CRUDs, uno para gestionar la facturación y la pasarela de pago y otro que permite gestionar las órdenes de las facturaciones.

Para empezar a trabajar antes hemos tenido que registrarnos en Authorize.net

5.7.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	10	0	0	0	0	0	0
Vista y Controlador	40	40	10	20	0	0	0
Modelo	30	20	30	10	0	0	0
Test	20	10	5	0	0	0	0
Total	110	70	45	30	0	0	0

Sprint 7



5.7.2. Gemas utilizadas

En este sprint hemos necesitado añadir las gemas:

- activemerchant, versión 1.56.0
- countries, versión 0.9.3
- country_select, versión 1.3.1

5.7.3. Ficheros de migración

Modelo y del fichero de migración de pedido:

```
ine@yii ~/projects/eshop1 $ rails generate model Order  
ine@yii ~/projects/eshop1 $ rails generate model OrderItem
```

Editamos pues los ficheros de migración:

Fichero: \db\migrate\20190506120007_create_orders.rb

```
class CreateOrders < ActiveRecord::Migration[5.1]  
  def change  
    create_table :orders do |t|  
      # contact information  
      t.string :email  
      t.string :phone_number  
      # shipping address  
      t.string :ship_to_first_name  
      t.string :ship_to_last_name  
      t.string :ship_to_address  
      t.string :ship_to_city  
      t.string :ship_to_postal_code  
      t.string :ship_to_country_code  
      # private fields  
      t.string :customer_ip  
      t.string :status  
      t.string :error_message  
      t.timestamps  
    end  
  end  
end
```

Fichero: \db\migrate\20190506120008_create_order_items.rb

```
class CreateOrderItems < ActiveRecord::Migration[5.1]
  def self.up
    create_table :order_items do |t|
      t.integer :game_id, :limit => 8
      t.integer :order_id, :limit => 8
      t.float :price
      t.integer :amount
      t.timestamps
    end

    say_with_time 'Adding foreing keys' do
      # Add foreign key reference to order_items table
      execute 'ALTER TABLE order_items ADD CONSTRAINT fk_order_items_orders
               FOREIGN KEY (order_id) REFERENCES orders(id) ON DELETE CASCADE'
    end
  end

  def self.down
    drop_table :order_items
  end
end
```

Para que se creen convenientemente las tablas en la base de datos:

ine@yii ~/projects/eshop1 \$ rake db:migrate

5.7.4. Modelos ORM

Los ficheros correspondiente al modelo quedarían de la siguiente forma:

Fichero: app / models / order . rb

```

class Order < ActiveRecord::Base
  require "active_merchant/billing/rails"

  attr_accessor :card_type, :card_number, :card_expiration_month, :card_expiration_year,
    :card_verification_value

  has_many :order_items
  has_many :games, :through => :order_items

  validates_presence_of :order_items,
    :message => '|Su carrito de la compra está vacío! ' +
      'Por favor, agregue al menos un juego antes de realizar el pedido.'
  validates_format_of :email, :with => /\A([^\s]+@[^\s]+\.(?=-a-zA-Z\d{2,})\Z/i
  validates_length_of :phone_number, :in => 7..20

  validates_length_of :ship_to_first_name, :in => 2..255
  validates_length_of :ship_to_last_name, :in => 2..255
  validates_length_of :ship_to_address, :in => 2..255
  validates_length_of :ship_to_city, :in => 2..255
  validates_length_of :ship_to_postal_code, :in => 2..255
  validates_length_of :ship_to_country_code, :in => 2..255

  validates_length_of :customer_ip, :in => 7..15
  validates_inclusion_of :status, :in => %w(open processed closed failed)

  validates_inclusion_of :card_type, :in => ['Visa', 'MasterCard', 'American Express', 'Discover'], :on => :create
  validates_length_of :card_number, :in => 13..19, :on => :create
  validates_inclusion_of :card_expiration_month, :in => %w(1 2 3 4 5 6 7 8 9 10 11 12), :on => :create
  validates_inclusion_of :card_expiration_year, :in => %w(2017 2018 2019 2020 2021 2022), :on => :create
  validates_length_of :card_verification_value, :in => 3..4, :on => :create

  def total
    sum = 0
    order_items.each do |item|
      sum += item.price * item.amount
    end
    sum
  end

  def amount
    sum = 0
    order_items.each do |item|
      sum += item.amount
    end
    sum
  end

  def process
    begin
      raise 'No puede procesarse de nuevo un pedido ya cerrado' if self.closed?
      active_merchant_payment
    rescue => e
      logger.error("Pedido #{id} fallido debido a una excepción: #{e}")
      self.error_message = "Excepción generada: #{e}"
      self.status = 'failed'
    end
    save!
    self.processed?
  end

  def active_merchant_payment
    ActiveMerchant::Billing::Base.mode = :test
    ActiveMerchant::Billing::AuthorizeNetGateway.default_currency = 'EUR'
    ActiveMerchant::Billing::AuthorizeNetGateway.wiredump_device = STDERR
    ActiveMerchant::Billing::AuthorizeNetGateway.wiredump_device.sync = true
    self.status = 'failed' # order status by default
  end
end

```

```

# the card verification value is also known as CVV2, CVC2, or CID
creditcard = ActiveMerchant::Billing::CreditCard.new(
  :brand      => card_type,
  :number     => card_number,
  :month      => card_expiration_month,
  :year       => card_expiration_year,
  :verification_value => card_verification_value,
  :first_name  => ship_to_first_name,
  :last_name   => ship_to_last_name
)

# buyer information
shipping_address = {
  :first_name => ship_to_first_name,
  :last_name  => ship_to_last_name,
  :address1   => ship_to_address,
  :city        => ship_to_city,
  :zip         => ship_to_postal_code,
  :country    => ship_to_country_code,
  :phone       => phone_number,
}

# order information
details = {
  :description  => 'INPlay gamestore purchase',
  :order_id     => self.id,
  :email         => email,
  :ip            => customer_ip,
  :billing_address => shipping_address,
  :shipping_address => shipping_address
}

if creditcard.valid? # validating the card automatically detects the card type
  gateway = ActiveMerchant::Billing::AuthorizeNetGateway.new( # use the test account
    :login      => '6kKAxW59f8',
    :password   => '6G4zaaR252g76Fuv'
    # the statement ":test = 'true'" tells the gateway not to process transactions
  )

  # Active Merchant accepts all amounts as integer values in cents
  response = gateway.purchase(self.total * 100, creditcard, details)

  if response.success?
    self.status = 'processed'
  else
    self.error_message = response.message
  end
else
  self.error_message = 'Tarjeta de crédito no válida'
end
end

def processed?
  self.status == 'processed'
end

def failed?
  self.status == 'failed'
end

def closed?
  self.status == 'closed'
end

def close
  self.status = 'closed'
  save!
end
end

```

Fichero: app / models / order_item . rb

```
class OrderItem < ApplicationRecord
  belongs_to :order
  belongs_to :game

  def validate
    errors.add(:amount, "debe ser uno o más") unless amount.nil? || amount > 0
    errors.add(:price, "debe ser un número positivo") unless price.nil? || price > 0.0
  end
end
```

5.7.5. Vistas y controladores

Para obtener ficheros de controladores de Checkout y admin/Order, ejecutar:

```
ine@gii ~/projects/eshop1 $ rails generate controller Checkout index submit_order thank_you
ine@gii ~/projects/eshop1 $ rails generate controller admin/order show index close
```

Editamos ambos controladores convenientemente:

Fichero: app / controllers / checkout_controller . rb

```
class CheckoutController < ApplicationController
  before_action :initialize_cart, :only => :index

  def index
    @order = Order.new
    @page_title = 'Facturación'
    if @cart.games.empty?
      flash[:notice] = '|Su carrito de la compra está vacío! ' +
                      'Por favor, agrégue al menos un juego antes de proceder a la facturación.'
      redirect_to :controller => 'catalog'
    end
  end

  def submit_order
    @cart = Cart.find(params[:cart][:id]) # Search the cart from the cart id hidden field of the form
    @order = Order.new(order_params)
    @order.ship_to_country_code = @order.ship_to_country_code.upcase
    @order.customer_ip = request.remote_ip
    @order.status = 'open'
    @page_title = 'Facturación'
    populate_order
  end
```

```

    if @order.save
      if @order.process
        flash[:notice] = 'Su pedido ha sido realizado y se procesará inmediatamente.'
        session[:order_id] = @order.id
        @cart.cart_items.destroy_all # empty shopping cart
        redirect_to :action => 'thank_you'
      else
        flash[:notice] = "Error al realizar el pedido '#{@order.error_message}'."
        render :action => 'index'
      end
    else
      render :action => 'index'
    end
  end

  def thank_you
    @page_title = 'Gracias.'
  end

  private

  def populate_order
    for cart_item in @cart.cart_items
      order_item = OrderItem.new(:game_id => cart_item.game_id,
                                 :price => cart_item.price,
                                 :amount => cart_item.amount)
      @order.order_items << order_item
    end
  end

  def order_params
    params.require(:order).permit(:email, :phone_number, :ship_to_first_name, :ship_to_last_name, :ship_to_address, :ship_to_city
      ,:card_expiration_month, :card_expiration_year, :card_number, :card_verification_value)
  end
end

```

Fichero: app / controllers / admin / order_controller . rb

```

class Admin::OrderController < Admin::AuthenticatedController
  def close
    order = Order.find(params[:id])
    order.close
    flash[:notice] = "El pedido ##{@order.id} ha sido cerrado."
    redirect_to :action => 'index'
  end

  def show
    @order = Order.find(params[:id])
    @page_title = "Mostrando pedido ##{@order.id}"
  end

  def index
    @status = params[:id]
    if @status.blank?
      conditions = nil
      @status = 'all'
      @page_title = 'Listando todos los pedidos'
    else
      conditions = {status: @status}
      @page_title = "Listando pedidos #{@status}"
    end
    @orders = Order.page(params[:page]).per(10).order('created_at DESC')
  end
end

```

```

    else
      conditions = "status = '#{@status}'"
      @estado = 'abierto' if @status == 'open'
      @estado = 'procesado' if @status == 'processed'
      @estado = 'cerrado' if @status == 'closed'
      @estado = 'fallido' if @status == 'failed'
      @page_title = "Listando pedidos #{@estado.pluralize}"
    end
  @orders = Order.where(conditions).paginate(:page => params[:page], :per_page => 10
) end
end

```

Seguidamente vamos a mostrar los ficheros correspondientes a las vistas:

Fichero: app / views / checkout / index . html . erb

```

<% if @order.errors.any? %>
  <div id="errorExplanation">
    <h2>Este pedido no se puede guardar debido a <%= pluralize(@order.errors.count, "error", "errores") %></h2>
    <ul>
      <% @order.errors.full_messages.each do |msg| %>
        <% if msg.include?('Order items') %>
          <li><%= msg.gsub('Order items ', '') %></li>
        <% else %>
          <li><%= msg %></li>
        <% end %>
      <% end %>
    </ul>
  </div>
<% end %>

<p><em>Su pedido se muestra en el carrito de la compra a la derecha.</em></p>
<%= form_tag :action => 'submit_order' do %>
  <div id="checkout">
    <fieldset>
      <legend>Información de contacto</legend>
      <p><label for="order_email">Correo electrónico</label><br/>
        <%= text_field :order, :email %></p>
      <p><label for="order_phone_number">Teléfono</label><br/>
        <%= text_field :order, :phone_number %></p>
    </fieldset>

    <fieldset>
      <legend>Dirección de envío</legend>
      <p><label for="order_ship_to_first_name">Nombre</label><br/>
        <%= text_field :order, :ship_to_first_name %></p>
      <p><label for="order_ship_to_last_name">Apellidos</label><br/>
        <%= text_field :order, :ship_to_last_name %></p>
      <p><label for="order_ship_to_address">Dirección</label><br/>
        <%= text_field :order, :ship_to_address %></p>
      <p><label for="order_ship_to_city">Ciudad</label><br/>
        <%= text_field :order, :ship_to_city %></p>
      <p><label for="order_ship_to_postal_code">Código postal</label><br/>
        <%= text_field :order, :ship_to_postal_code %></p>
      <p><label for="order_ship_to_country_code">País</label><br/>
        <%= country_select(:order, :ship_to_country_code, priority_countries: ['ES']) %></p>
    </fieldset>
  </div>
<% end %>

```

```

<fieldset>
  <legend>Información de facturación</legend>
  <p><label for="order_card_type">Tipo de tarjeta de crédito</label><br/>
  <select name="order[card_type]" id="order_card_type">
    <%= options_for_select(["Visa", "MasterCard", "American Express", "Discover"], @order.card_type) %>
  </select></p>
  <p><label for="order_card_expiration_month">Fecha de caducidad</label><br/>
  <select name="order[card_expiration_month]">
    <%= options_for_select(%w(1 2 3 4 5 6 7 8 9 10 11 12), @order.card_expiration_month) %>
  </select>
  <select name="order[card_expiration_year]">
    <%= options_for_select(%w(2017 2018 2019 2020 2021 2022), @order.card_expiration_year) %>
  </select></p>
  <p><label for="order_card_number">Número de tarjeta</label><br/>
  <%= text_field :order, :card_number %></p>
  <p>
    <label for="order_card_verification_value">
      <abbr title="Card Verification Value">CVV</abbr>
      <abbr title="Card Validation Check">CVC</abbr>
    </label><br/>
    <%= text_field :order, :card_verification_value %>
  </p>
</fieldset>

<div class="field">
  <%= hidden_field :cart, :id %>
</div>

<p><%= submit_tag "Realizar pedido" %></p>
</div>
<% end %>

```

Fichero: app / views / checkout / thank_you . html . erb

Para futuras referencias use el numero de factura <%= session[:order_id] %>

Fichero: app / views / admin / order / navigation . html . erb

```

<p>
  <strong>Listar: <%= link_to 'todos', :action => 'index', :id => '' %>,
  <%= link_to 'abiertos', :action => 'index', :id => 'open' %>,
  <%= link_to 'procesados', :action => 'index', :id => 'processed' %>,
  <%= link_to 'cerrados', :action => 'index', :id => 'closed' %>,
  <%= link_to 'fallidos', :action => 'index', :id => 'failed' %></strong>
</p>

```

Fichero: app / views / admin / order / index . html . erb

```
<% if @orders == [] %>
<% if @status == 'all' %>
  <h2><%= "No hay pedidos." %></h2>
<% else %>
  <h2><%= "No hay pedidos con el estado '#{@estado}'. " %></h2>
<% end %>
<% else %>
<table>
  <tr>
    <th>ID</th>
    <th>Estado</th>
    <th>Precio total</th>
    <th>Cantidad</th>
    <th>Creado el</th>
    <th>Actualizado el</th>
    <th></th>
  </tr>
<% for order in @orders %>
  <% estado = 'procesado' if order.status == 'processed'
     estado = 'fallido' if order.status == 'failed'
     estado = 'abierto' if order.status == 'open'
     estado = 'cerrado' if order.status == 'closed' %>
  <tr>
    <td align="center"><%= order.id %></td>
    <td align="center"><%= estado.capitalize %></td>
    <td align="center"><%= order.total %></td>
    <td align="center"><%= order.amount %></td>
    <td align="center"><%= order.created_at.strftime("%d-%m-%Y %H:%M") %></td>
    <td align="center"><%= order.updated_at.strftime("%d-%m-%Y %H:%M") %></td>
    <td><%= link_to 'Mostrar', :action => 'show', :id => order %></td>
  </tr>
<% end %>
</table>

<% if @orders.total_pages > 1 %>
  <br/>
  <%= 'Ver página:' %>
<% end %>

<%= will_paginate @orders, :page_links => true, :link_separator => ' ', :container => false,
                      :previous_label => '', :next_label => '' %>
<p></p>
<% end %>

<%= render :partial => 'navigation' %>
```

Fichero: app / views / admin / order / show . html . erb

```
<h2>Información de contacto</h2>
<dl>
  <dt>Correo electrónico</dt>
  <dd><%= @order.email %></dd>
  <dt>Teléfono</dt>
  <dd><%= @order.phone_number %></dd>
</dl>

<h2>Dirección de envío</h2>
<dl>
  <dt>Nombre</dt>
  <dd><%= @order.ship_to_first_name %></dd>
  <dt>Apellidos</dt>
  <dd><%= @order.ship_to_last_name %></dd>
  <dt>Dirección</dt>
  <dd><%= @order.ship_to_address %></dd>
  <dt>Ciudad</dt>
  <dd><%= @order.ship_to_city %></dd>
  <dt>Código postal</dt>
  <dd><%= @order.ship_to_postal_code %></dd>
  <dt>País</dt>
  <dd><%= ISO3166::Country.find_country_by_alpha2(@order.ship_to_country_code).name %></dd>
</dl>

<h2>Detalles</h2>
<% for item in @order.order_items %>
  <%= link_to item.game.name, :controller => 'game', :action => 'show', :id => item.game.id %>
  <%= pluralize(item.amount, "juego", "juegos") %>, <%= item.price * item.amount %> € <br/>
<% end %>

<p><strong>Total: <%= @order.total %> €</strong></p>

<h2>Estado</h2>
<dl>
  <dt>Estado</dt>
  <% estado = 'procesado' if @order.status == 'processed'
    estado = 'fallido' if @order.status == 'failed'
    estado = 'abierto' if @order.status == 'open'
    estado = 'cerrado' if @order.status == 'closed' %>
  <dd><%= estado.capitalize %></dd>
  <% if @order.failed? %>
    <dt>Error</dt>
    <dd><%= @order.error_message %></dd>
  <% end %>
</dl>

<% if !@order.closed? %> <p></p> <% end %>
<%= button_to 'Cerrar pedido', {:action => 'close', :id => @order},
  data: { confirm: "{Está seguro de que quiere cerrar el pedido ##{@order.id}?" } if @order.processed? %>
<%= render :partial => 'navigation' %>
```

5.7.6. Test

Vamos con el test del modelo de pedido:

Fichero: test / models / order_test . rb

```
require 'test_helper'

class OrderTest < ActiveSupport::TestCase
  test "create_valid_order" do
    order = Order.new(
      # Contact information
      :email => 'email@email.com',
      :phone_number => '666112233',
      # Shipping address
      :ship_to_first_name => 'Firstname',
      :ship_to_last_name => 'Lastname',
      :ship_to_address => 'Address',
      :ship_to_city => 'City',
      :ship_to_postal_code => '00000',
      :ship_to_country_code => 'ES',
      # Billing information
      :card_type => 'Visa',
      :card_number => '4007000000027',
      :card_expiration_month => '12',
      :card_expiration_year => '2022',
      :card_verification_value => '000'
    )

    # Private information
    order.customer_ip = '127.0.0.1'
    order.status = 'open'

    order.order_items << OrderItem.new(:game_id => 1, :price => 155.25, :amount => 3)

    assert order.save
    assert order.process
    order.reload
    assert_equal 1, order.order_items.size
    assert_equal 155.25, order.order_items[0].price
    assert_equal order.status, 'processed'
    order.close
    assert order.closed?
  end

  test "validations" do
    order = Order.new
    assert_equal false, order.save
    assert_equal 16, order.errors.size

    # An order must have at least one order item
    assert order.errors[:order_items]
    # Contact information
    assert order.errors[:email]
    assert order.errors[:phone_number]
```

```

# Shipping address
assert order.errors[:ship_to_first_name]
assert order.errors[:ship_to_last_name]
assert order.errors[:ship_to_address]
assert order.errors[:ship_to_city]
assert order.errors[:ship_to_postal_code]
assert order.errors[:ship_to_country_code]
# Billing information
assert order.errors[:card_type]
assert order.errors[:card_number]
assert order.errors[:card_expiration_month]
assert order.errors[:card_expiration_year]
assert order.errors[:card_verification_value]
# Private information
assert order.errors[:customer_ip]
assert order.errors[:status]
end
end

```

Crearemos ahora el fichero para el test de Integración de facturación:

```
ine@gii ~/projects/eshop1 $ rails generate integration_test Checkout
```

Y modificamos el fichero correspondiente:

Fichero: test/integration/checkout_test.rb

```

require 'test_helper'

class CheckoutTest < ActionDispatch::IntegrationTest
  fixtures :developers, :suppliers, :games

  test "empty_cart_shows_error_message" do
    get '/checkout'
    assert_response :redirect
    assert_redirected_to :controller => 'catalog'
    assert_equal flash[:notice], '¡Su carrito de la compra está vacío! ' +
      'Por favor, agrégue al menos un juego antes de proceder a la facturación.'
  end

  test "submitting_order" do
    post '/cart/add', :params => { :id => 1 }
    get '/checkout'
    assert_response :success
    assert_select 'legend', 'Información de contacto'
    assert_select 'legend', 'Dirección de envío'
    assert_select 'legend', 'Información de facturación'

    post '/checkout/submit_order', :params => { :cart => { :id => Cart.last.id }, :order => {
      :first_name => 'John',
      :last_name => 'Doe',
      :address => '123 Main Street',
      :city => 'Anytown',
      :state => 'CA',
      :zip => '12345',
      :country => 'USA',
      :phone => '555-1234',
      :email => 'john.doe@example.com'
    } }
  end
end

```

```

# Contact information
:email => 'email@email.com',
:phone_number => '666112233',
# Shipping address
:ship_to_first_name => 'Firstname',
:ship_to_last_name => 'Lastname',
:ship_to_address => 'Address',
:ship_to_city => 'City',
:ship_to_postal_code => '00000',
:ship_to_country_code => 'Country',
# Billing information
:card_type => 'Visa',
:card_number => '4007000000027',
:card_expiration_month => '12',
:card_expiration_year => '2022',
:card_verification_value => '000'
)}

assert_response :redirect
assert_redirected_to '/checkout/thank_you'
end
end

```

Vamos a ejecutar primero el del modelo de pedido:

```
ine@yii ~/projects/eshop1 $ rake test TEST=test/models/order_test.rb
```

Y ahora el test de integración de facturación:

```
ine@yii ~/projects/eshop1 $ rake test TEST=test/integration/checkout_test.rb
```

5.7.7. Dificultades encontradas

Hemos experimentado algunos problemas con la plataforma Authorize.net, por ser algo totalmente nuevo para nosotros, y entender el funcionamiento de la pasarela de pago, con la ayuda del profesor hemos podido ir solventando dichos problemas.

5.7.8. Objetivos alcanzados

Se ha conseguido con éxito la facturación y la gestión de los pedidos por parte del administrador.

5.6. Sprint 8 (Autenticación)

Implementación de la autenticación del administrador, que será el único capaz de administrar los artículos, fabricantes, proveedores y pedidos.

5.8.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	0	0	0	0	0	0	0
Vista y Controlador	40	30	30	30	0	0	0
Modelo	80	10	20	10	0	0	0
Test	10	10	10	10	0	0	0
Total	130	50	60	50	0	0	0

Sprint 8



5.8.2. Gemas utilizadas

En este sprint hemos añadido:

- authlogic, versión 3.4.6: gestiona el acceso mediante contraseña.

5.8.3. Ficheros de migración

Vamos a introducir el comando que permite crear las plantillas del modelo, del test del modelo y del fichero de migración de User y UserSession:

```
ine@gii ~/projects/eshop1 $ rails generate model User
ine@gii ~/projects/eshop1 $ rails generate model UserSession
```

Editamos pues los ficheros de migración:

Fichero: \db\migrate\20190506120006_create_users.rbb

```
class CreateUsers < ActiveRecord::Migration[5.1]
  def change
    create_table :users do |t|
      t.string :name, :null => false, :unique => true
      t.string :login, :null => false, :unique => true
      t.string :email, :null => false, :unique => true
      t.string :encrypted_password, :null => false
      t.string :password_salt, :null => false
      t.string :persistence_token, :null => false
      t.string :perishable_token, :null => false # optional, used for reset password functionality

      #t.string :single_access_token, :null => false # optional, see Authlogic::Session::Params

      # magic fields (all optional, see Authlogic::Session::MagicColumns)
      t.integer :login_count, :null => false, :default => 0
      t.integer :failed_login_count, :null => false, :default => 0
      t.datetime :last_request_at
      t.datetime :current_login_at
      t.datetime :last_login_at
      t.string :current_login_ip
      t.string :last_login_ip
      t.timestamps
    end
  end
end
```

Para que se creen las tablas en nuestra base de datos:

```
ine@gii ~/projects/eshop1 $ rake db:migrate
```

5.8.4. Modelos ORM

Los ficheros del modelo serían así:

Fichero: app / models / user . rb

```
class UserController < ApplicationController
  before_action :require_user, :only => [:show, :edit, :update]

  def new
    @user = User.new
    @page_title = 'Crear nueva cuenta'
    if current_user
      flash[:notice] = 'Sólo se puede crear una cuenta.'
      redirect_to :controller => 'about', :action => 'index'
    else
      # only when there are no accounts it allows to create a new one, unique in the system
      redirect_to :controller => 'user_sessions', :action => 'new' unless User.count == 0
    end
  end

  def create
    @user = User.new(user_params)
    if @user.save # the new user has been logged in automatically
      flash[:notice] = "Cuenta #{@user.name} creada correctamente. Sesión iniciada."
      redirect_to :action => 'show'
    else
      @page_title = 'Crear nueva cuenta'
      render :action => :new
    end
  end

  def edit
    @user = current_user
    @page_title = 'Editar cuenta'
  end

  def update
    @user = current_user
    if @user.update_attributes(user_params)
      flash[:notice] = "Cuenta #{@user.name} actualizada correctamente."
      redirect_to :action => 'show'
    else
      @page_title = 'Editar cuenta'
      render :action => 'edit'
    end
  end

  def show
    @user = current_user
    @page_title = @user.name
  end
```

```
private
  def user_params
    params.require(:user).permit(:name, :login, :email, :password, :password_confirmation)
  end
end
```

Fichero: app / models / user_session . rb

```
class UserSession < Authlogic::Session::Base
  logout_on_timeout true # default if false
end
```

5.8.5. Vistas y controladores

Para la generación de los ficheros de controladores de User, UserSession y admin/Authenticated, ejecutamos los siguientes comandos:

```
ine@gii ~/projects/eshop1 $ rails generate controller user new create edit update show
ine@gii ~/projects/eshop1 $ rails generate controller user_sessions new create destroy
ine@gii ~/projects/eshop1 $ rails generate controller admin/authenticated
```

Editamos los controladores:

Fichero: app / controllers / user_controller . rb

```
class UserController < ApplicationController
  before_action :require_user, :only => [:show, :edit, :update]

  def new
    @user = User.new
    @page_title = 'Crear nueva cuenta'
    if current_user
      flash[:notice] = 'Sólo se puede crear una cuenta.'
      redirect_to :controller => 'about', :action => 'index'
```

```

    else
      # only when there are no accounts it allows to create a new one, unique in the system
      redirect_to :controller => 'user_sessions', :action => 'new' unless User.count == 0
    end
  end

  def create
    @user = User.new(user_params)
    if @user.save # the new user has been logged in automatically
      flash[:notice] = "Cuenta #{@user.name} creada correctamente. Sesión iniciada."
      redirect_to :action => 'show'
    else
      @page_title = 'Crear nueva cuenta'
      render :action => :new
    end
  end

  def edit
    @user = current_user
    @page_title = 'Editar cuenta'
  end

  def update
    @user = current_user
    if @user.update_attributes(user_params)
      flash[:notice] = "Cuenta #{@user.name} actualizada correctamente."
      redirect_to :action => 'show'
    else
      @page_title = 'Editar cuenta'
      render :action => 'edit'
    end
  end

  def show
    @user = current_user
    @page_title = @user.name
  end

  private
  def user_params
    params.require(:user).permit(:name, :login, :email, :password, :password_confirmation)
  end
end

```

Fichero: app / controllers / user_sessions_controller . rb

```

class UserSessionsController < ApplicationController
  before_action :require_no_user, :only => [:new, :create]

```

```

def new
  @user_session = UserSession.new
  @page_title = 'Inicio de sesión'
end

def create
  @user_session = UserSession.new(user_session_params)
  @user_session.remember_me = false # just in case
  if @user_session.save
    flash[:notice] = "Inicio de sesión correcto."
    redirect_back_or_default :controller => '/admin/artist', :action => :index
# default login route
  else
    render :action => :new
  end
end

def destroy
  if current_user_session # only for an authenticated user
    current_user_session.destroy
    flash[:notice] = "Fin de sesión correcto."
  end
  redirect_to :controller => :catalog, :action => :index # logout route
end

private
def user_session_params
  params.require(:user_session).permit(:login, :password, :remember_me)
end
end

```

Fichero: app / controllers / admin / authenticated_controller . rb

```

class Admin::AuthenticatedController < ApplicationController
  before_action :require_user
end

```

Como las entidades administradas por el administrador necesitan de su autenticación,

de este deberemos modificar algunas líneas de los siguientes controladores:

Fichero: app / controllers / admin / game_controller . rb

```
Admin::GameController < Admin::AuthenticatedController
def new
= Game.new
= 'Crear nuevo juego'
end

def create
= Game.new(game_params)
if @game.save
= "Juego #{@game.name} creado correctamente."
'index'
else
= 'Crear nuevo juego'
'new'
end
end

def edit
= Game.find(params[:id])
= 'Editar juego'
end

def update
= Game.find(params[:id])
if @game.update_attributes(game_params)
= "Juego #{@game.name} actualizado correctamente."
'show', :id => @game
else
= 'Editar juego'
'edit'
end
end

def destroy
= Game.find(params[:id])
= "Juego #{@game.name} eliminado correctamente."
'index'
end
```

```

def show
= Game.find(params[:id])
= @game.name
end

def index
= params[:sort_by]
= Game.order(sort_by).paginate(:page => params[:page], :per_page => 5
⌘ 'Listado de juegos'
end

private

def load_data
= Supplier.all
= Developer.all
end

def game_params
end

```

Fichero: app / controllers / admin / developer_controller . rb

```

class Admin::DeveloperController < Admin::AuthenticatedController
  def new
    @developer = Developer.new
    @page_title = 'Crear nuevo desarrollador'
  end

  def create
    @developer = Developer.new(developer_params)
    if @developer.save
      flash[:notice] = "Desarrollador #{@developer.name} creado correctamente."
    redirect_to :action => 'index'
  end

```

```

    else
      @page_title = 'Crear nuevo desarrollador'
      render :action => 'new'
    end
  end

  def edit
    @developer = Developer.find(params[:id])
    @page_title = 'Editar desarrollador'
  end

  def update
    @developer = Developer.find(params[:id])
    if @developer.update_attributes(developer_params)
      flash[:notice] = "Desarrollador #{@developer.name} actualizada correctamente."
      redirect_to :action => 'show', :id => @developer
    else
      @page_title = 'Editar desarrollador'
      render :action => 'edit'
    end
  end

  def show
    @developer = Developer.find(params[:id])
    @page_title = @developer.name
  end

  def index
    @developers = Developer.all
    @page_title = 'Listado de desarrolladores'
  end

  def destroy
    @developer = Developer.find(params[:id])
    @developer.destroy
    flash[:notice] = "Desarrollador #{@developer.name} eliminado correctamente."
    redirect_to :action => 'index'
  end

  private
    def developer_params
      params.require(:developer).permit(:name)
    end
  end

```

Fichero: app / controllers / admin / supplier_controller . rb

```
class Admin::SupplierController < Admin::AuthenticatedController
  def new
    @supplier = Supplier.new
    @page_title = 'Crear nuevo proveedor'
  end

  def create
    @supplier = Supplier.new(supplier_params)
    if @supplier.save
      flash[:notice] = "Proveedor #{@supplier.name} creado correctamente."
      redirect_to :action => 'index'
    else
      @page_title = 'Crear nuevo proveedor'
      render :action => 'new'
    end
  end

  def edit
    @supplier = Supplier.find(params[:id])
    @page_title = 'Editar proveedor'
  end

  def update
    @supplier = Supplier.find(params[:id])
    if @supplier.update_attributes(supplier_params)
      flash[:notice] = "Proveedor #{@supplier.name} actualizado correctamente."
      redirect_to :action => 'show', :id => @supplier
    else
      @page_title = 'Editar proveedor'
      render :action => 'edit'
    end
  end

  def destroy
    @supplier = Supplier.find(params[:id])
    @supplier.destroy
  end
```

```

    flash[:notice] = "Proveedor #{@supplier.name} eliminado correctamente."
    redirect_to :action => 'index'
  end

  def show
    @supplier = Supplier.find(params[:id])
    @page_title = @supplier.name
  end

  def index
    @suppliers = Supplier.all
    @page_title = 'Listado de proveedores'
  end

  private
  def supplier_params
    params.require(:supplier).permit(:first_name, :last_name)
  end
end

```

Fichero: app / controllers / admin / order_controller . rb

```

class Admin::OrderController < Admin::AuthenticatedController
  def close
    order = Order.find(params[:id])
    order.close
    flash[:notice] = "El pedido ##{@order.id} ha sido cerrado."
    redirect_to :action => 'index'
  end

  def show
    @order = Order.find(params[:id])
    @page_title = "Mostrando pedido ##{@order.id}"
  end

  def index
    @status = params[:id]
    if @status.blank?

```

```

conditions = nil
@status = 'all'
@page_title = 'Listando todos los pedidos'
else
  conditions = "status = '#{@status}'"
  @estado = 'abierto' if @status == 'open'
  @estado = 'procesado' if @status == 'processed'
  @estado = 'cerrado' if @status == 'closed'
  @estado = 'fallido' if @status == 'failed'
  @page_title = "Listando pedidos #{@estado.pluralize}"
end
@orders =
Order.where(conditions).paginate(:page => params[:page], :per_page => 10)
end
end

```

Ahora debemos modificar el application.html.erb

Fichero: app / views / layouts / application . html . erb

```

<!DOCTYPE html>
<html>
<head>
  <title><%= @page_title || 'INEPlay' %></title>
  <%= csrf_meta_tags %>
  <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track': 'reload' %>
  <%= javascript_include_tag 'application', 'data-turbolinks-track': 'reload' %>
</head>

<body>
  <div id="header">
    <h1 id="logo">INEPlay&trade;</h1>
    <h2 id="slogan">Videojuegos sobre Railes</h2>
    <% if current_user %>
      <p id="loginlogout">
        Identificado como <%= current_user.login %>
        (<%= link_to "Editar cuenta", :controller => '/user', :action => :show %>)
        <br/>
        (<%= link_to "Cerrar sesión", :controller => '/user_sessions', :action => :destroy %>)
      </p>
    <% else %>
      <p id="loginlogout">
        <% if User.count == 0 %>
          (<%= link_to "Crear nueva cuenta", :controller => '/user', :action => :new %>)
        <% else %>

```

```

<% else %>
    (<%= link_to "Iniciar sesión", :controller => '/user_sessions', :action => :new %>)
<% end %>
</p>
<% end %>
</div>

<div id="menu">
<ul>
<li><a href="/admin/developer">Desarrolladores</a>&nbsp;|&nbsp;</li>
<li><a href="/admin/supplier">Proveedores</a>&nbsp;|&nbsp;</li>
<li><a href="/admin/game">Juegos</a>&nbsp;|&nbsp;</li>
<li><a href="/admin/order">Pedidos</a>&nbsp;|&nbsp;</li>
<li><a href="/">Catálogo</a>&nbsp;|&nbsp;</li>
<li><a href="/about">Sobre INEPlay</a>&nbsp;</li>
</ul>
</div>

<div id="content">
<h1><%= @page_title if @page_title %></h1>
<% if flash[:notice] %>
    <div id="notice"><%= flash[:notice] %></div>
<% end %>
<%= yield %>
</div>

<% if @cart %>
    <div id="shopping_cart"><%= render :partial => 'cart/cart' %></div>
<% end %>

<div id="footer">
    &copy; 2015-2019 INEPlay
</div>
</body>
</html>

```

Y después, las vistas:

Fichero: app / views / user / form . html . erb

```

<% if @user.errors.any? %>
<div id="errorExplanation">
<h2>Esta cuenta no se puede guardar debido a <%= pluralize(@user.errors.count, "error", "errores") %></h2>
<ul>
    <% @user.errors.full_messages.each do |msg| %>
        <li><%= msg %></li>
    <% end %>
</ul>
</div>
<% end %>

<div class="field">
<p><label for="user_name">Nombre</label><br/>
<%= text_field 'user', 'name' %></p>
</div>

```

```

<div class="field">
  <p><label for="user_name">Nombre</label><br/>
  <%= text_field 'user', 'name' %></p>
</div>

<div class="field">
  <p><label for="user_login">Usuario</label><br/>
  <%= text_field 'user', 'login' %></p>
</div>

<div class="field">
  <p><label for="user_email">Correo electrónico</label><br/>
  <%= text_field 'user', 'email' %></p>
</div>

<div class="field">
  <p><label for="user_password">Contraseña</label><br/>
  <%= password_field 'user', 'password' %></p>
</div>

<div class="field">
  <p><label for="user_password_confirmation">Confirmación de contraseña</label><br/>
  <%= password_field 'user', 'password_confirmation' %></p>
</div>

```

Fichero: app / views / user / new . html . erb

```

<%= form_tag :action => 'create' do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Crear cuenta' %>
<% end %>

```

Fichero: app / views / user / edit . html . erb

```

<%= form_tag :action => 'update', :id => @user do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Actualizar cuenta' %>
<% end %>

<%= link_to 'Atrás', :action => 'show' %>

```

Fichero: app / views / user / show . html . erb

```
<dl>
  <dt>Nombre</dt>
  <dd><%= @user.name %></dd>

  <dt>Usuario</dt>
  <dd><%= @user.login %></dd>

  <dt>Correo electrónico</dt>
  <dd><%= @user.email %></dd>

  <dt>Número de sesiones</dt>
  <dd><%= @user.login_count %></dd>

  <dt>Última petición</dt>
  <dd><%= @user.last_request_at %></dd>

  <% if @user.last_login_at %>
    <dt>Sesión anterior</dt>
    <dd><%= @user.last_login_at %></dd>
  <% end %>

  <dt>Sesión actual</dt>
  <dd><%= @user.current_login_at %></dd>

  <% if @user.last_login_ip %>
    <dt>IP de sesión anterior</dt>
    <dd><%= @user.last_login_ip %></dd>
  <% end %>

  <dt>IP de sesión actual</dt>
  <dd><%= @user.current_login_ip %></dd>
</dl>

<%= link_to 'Editar', :action => 'edit' %>
```

Fichero: app / views / user_sessions / form . html . erb

```
<% if @user_session.errors.any? %>
  <div id="errorExplanation">
    <h2>Este usuario no puede iniciar sesión debido a <%= pluralize(@user_session.errors.count, "error", "errores") %></h2>
    <ul>
      <% @user_session.errors.full_messages.each do |msg| %>
        <li><%= msg %></li>
      <% end %>
    </ul>
  </div>
<% end %>

<div class="field">
  <p><label for="login">Usuario</label><br/>
  <%= text_field 'user_session', 'login' %></p>
</div>

<div class="field">
  <p><label for="password">Contraseña</label><br/>
  <%= password_field 'user_session', 'password' %></p>
</div>

<!-- <div class="check_box">
  <p><label for="remember_me">Recordarme</label><br/>
  <%= check_box 'user_session', 'remember_me' %></p>
</div> -->
```

Fichero: app / views / user_sessions / new . html . erb

```
<%= form_tag :action => 'create' do %>
  <%= render :partial => 'form' %>
  <%= submit_tag 'Iniciar sesión' %>
<% end %>

<%= link_to 'No me acuerdo de mi contraseña', :controller => 'password_reset', :action => 'new' %>
```

5.8.6. Test

Se han realizado las siguientes pruebas de integración:

```
ine@gii ~/projects/eshop1 $ rails generate integration_test authentication
ine@gii ~/projects/eshop1 $ rails generate integration_test User
```

Editamos los ficheros:

Fichero: test / integration / authentication_test . rb

```
require 'test_helper'

class AuthenticationTest < ActionDispatch::IntegrationTest

  def setup
    User.create(:name => 'George Smith',
               :login => 'george',
               :email => 'george@emporium.com',
               :password => 'cheetah',
               :password_confirmation => 'cheetah')
  end

  test "successful_login" do
    george = new_session_as(:george)
    george.tries_to_go_to_admin
    george.logs_in_succefully("george", "cheetah")
  end

  test "failed_login" do
    harry = new_session_as(:harry)
    harry.tries_to_go_to_admin
    harry.fails_login("harry", "micky")
  end

  private

  module AuthenticationTestDSL
    include ERB::Util
    attr_writer :name

    def tries_to_go_to_admin
      get '/admin/game/new'
      assert_response :redirect
      assert_redirected_to '/user_sessions/new'
    end

    def logs_in_succefully(login, password)
      post_login(login, password)
      assert_response :success
      assert_redirected_to '/admin/game/new'
    end

    def fails_login(login, password)
      post_login(login, password)
      assert_response :success
      assert_template 'user_sessions/new'
      assert_select 'div#content' do
        assert_select 'div#errorExplanation'
        assert_select 'li', 'Login is not valid'
      end
    end
  end
end
```

```

    end
end

private

def post_login(login, password)
  post '/user_sessions/create', :params => { :user_session => { :login => login, :password => password } }
end

def new_session_as(name)
  open_session do |session|
    session.extend(AuthenticationTestDSL)
    session.name = name
    yield session if block_given?
  end
end
end

```

Fichero: test / integration / user_test . rb

```

require 'test_helper'

class UserTest < ActionDispatch::IntegrationTest

  def setup
  end

  test "user_account" do
    george = new_session_as(:george)
    user_account = george.create_user_account(:user => { :name => 'George Smith', :login => 'george',
                                                       :email => 'george@emporium.com', :password => 'gold',
                                                       :password_confirmation => 'gold' })
    george.shows_user_account user_account
    george.edits_user_account(user_account, :user => { :name => 'George Jackson', :login => 'george',
                                                       :email => 'george@emporium.com', :password => 'silver',
                                                       :password_confirmation => 'silver' })
  end

  private

  module UserTestDSL
    include ERB::Util
    attr_writer :name

    def creates_user_account(parameters)
      user_name = parameters[:user][:name]
      get '/user/new'
      assert_response :success
      assert_template 'user/new'
      assert_select 'div#content' do
        assert_select 'h1', 'Crear nueva cuenta'
        assert_select 'input#user_name'
      end
      post '/user/create', :params => parameters
      assert_response :redirect
      follow_redirect!
      assert_response :success
      assert_template 'user/show'
      assert_equal flash[:notice], "Cuenta #{user_name} creada correctamente. Sesión iniciada."
      assert_select 'div#notice', "Cuenta #{user_name} creada correctamente. Sesión iniciada."
    end
  end
end

```

```

    return User.find_by_login(parameters[:user][:login])
end

def shows_user_account(user_account)
  get "/user/show/?id=#{user_account.id}"
  assert_response :success
  assert_template 'user/show'
  assert_select 'div#content' do
    assert_select 'h1', user_account.name
    assert_select 'dt', 'Nombre'
    assert_select 'dd', user_account.name
  end
end

def edits_user_account(user_account, parameters)
  get "/user/edit?id=#{user_account.id}"
  assert_response :success
  assert_template 'user/edit'
  assert_select 'div#content' do
    assert_select 'h1', 'Editar cuenta'
    assert_select 'input#user_name'
  end
  post "/user/update?id=#{user_account.id}", :params => parameters
  assert_response :redirect
  follow_redirect!
  assert_response :success
  assert_template 'user/show'
  user_name = parameters[:user][:name]
  assert_equal flash[:notice], "Cuenta #{user_name} actualizada correctamente."
  assert_select 'div#notice', "Cuenta #{user_name} actualizada correctamente."
end

def new_session_as(name)
  open_session do |session|
    session.extend(UserTestDSL)
    session.name = name
    yield session if block_given?
  end
end
end

```

Los ejecutamos:

```
ine@gii ~/projects/eshop1 $ rake test TEST=test/integration/authentication_test.rb
```

5.8.7. Dificultades encontradas

Hemos experimentado algunos problemas hasta que nos dimos cuenta que tuvimos que hacer alguna ligera modificación en la primera línea de los controladores que están dentro del directorio admin.

5.8.8. Objetivos alcanzados

El administrador ahora es el único que puede eliminar, modificar y añadir juegos, proveedores y desarrolladores.

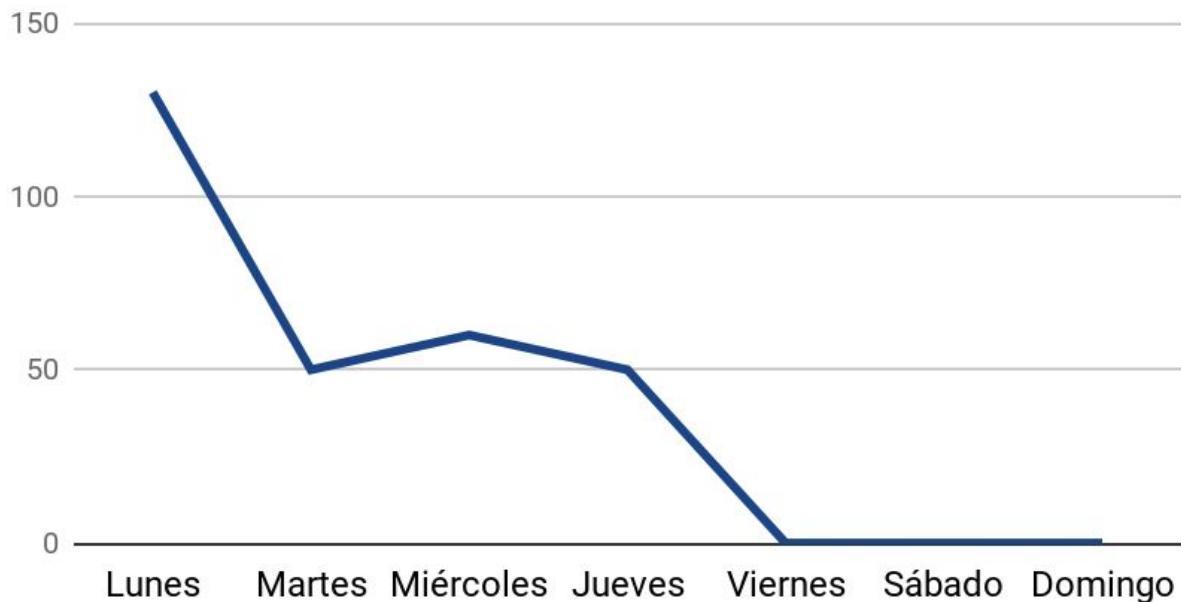
5.9. Sprint 9 (RSS y AJAX)

Vamos a añadir RSS (Really Simple Syndication) y AJAX (Asynchronous JavaScript And XML) a nuestra tienda. Integramos su función en las operaciones del carrito de la compra sin que se necesite actualizar la página entera

5.9.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	0	0	0	0	0	0	0
Vista y Controlador	40	30	30	30	0	0	0
Modelo	80	10	20	10	0	0	0
Test	10	10	10	10	0	0	0
Total	130	50	60	50	0	0	0

Sprint 9



5.9.2. Gemas utilizadas

Hemos añadido estas gemas:

- jquery-rails, versión 4.0.5: Biblioteca jQuery.
- jquery-ui-rails, versión 4.2.1: Animaciones de jQuery.

5.9.3. Ficheros de migración

5.9.4. Modelos ORM

No es necesario añadir código una vez comprobemos que todas las operaciones en cart.rb están correctamente diseñadas.

5.9.5. Vistas y controladores

Modificamos los siguientes archivos para añadir RSS al Catálogo:

Fichero: app / controllers / catalog_controller . rb

```
class CatalogController < ApplicationController
  before_action :initialize_cart, :except => :show
  # before_action :require_no_user

  def show
    @game = Game.find(params[:id])
    @page_title = @game.name
  end

  def index
    @games = Game.order("games.id desc").includes(:suppliers, :developer).paginate(:page => params[:page], :per_page => 5)
    @page_title = 'Catálogo'
  end

  def latest
    @games = Game.latest 5 # invoques "latest" method to get the five latest games
    @page_title = 'Últimos juegos'
  end

  def rss
    latest
    render :layout => false
    response.headers["Content-Type"] = "application/xml; version=1.0; charset=utf-8"
  end

  def search
    if params[:query] && !params[:query].empty?
      uppercase_query = params[:query].to_s.upcase
      if params[commit] == 'By title'
        @games = Game.search_by_title{uppercase_query}
      else
        @games = Game.search_by_author{uppercase_query}
      end
      flash.now[:notice] = ' No se han encontrado juegos.' unless @game.size > 0
    end
    @page_title = 'Search'
  end
end
```

Fichero: app / views / catalog / rss . builder

```

xml.instruct!
xml.rss "version" => "2.0", "xmlns:dc" =>
"http://purl.org/dc/elements/1.1/" do
xml.channel do
xml.title @page_title
xml.description "INEPlay: Games for gamers"
xml.link url_for :action => 'index', :only_path => false
xml.language "en-us"
xml.ttl "60"

for game in @games do
xml.item do
xml.title game.name
xml.description "#{game.name} by #{game.supplier_names}"
xml.link url_for :action => "show", :id => game,
:only_path => false
xml.guid url_for :action => "show", :id => game,
:only_path => false
xml.pubDate game.created_at.to_s :long
xml.supplier game.supplier_names
end
end
end
end

```

Realizamos cambios en estos ficheros para añadir la funcionalidad AJAX:

Fichero: app / controllers / cart_controller . rb

```

class CartController < ApplicationController
before_action :initialize_cart

def add
  @game = Game.find params[:id]
  @page_title = 'Añadir artículo'
  respond_to do |format|
    format.js { @item = @cart.add params[:id]
      flash.now[:cart_notice] = "Añadido <em>#{@item.game.name}</em>."
      render :controller => 'cart',
             :action => 'add_with_ajax' }
    format.html { if request.post?
      @item = @cart.add params[:id]
      flash[:cart_notice] = "Añadido <em>#{@item.game.name}</em>."
      redirect_to :controller => 'catalog'
    else
      render :controller => 'cart', :action => 'add', :template => 'cart/add'
    end }
  end
end

def remove
  @game = Game.find params[:id]
  @page_title = 'Eliminar artículo'
  respond_to do |format|
    format.js { @item = @cart.remove params[:id]
      flash.now[:cart_notice] = "Eliminado <em>#{@item.game.name}</em>."
      render :controller => 'cart',
             :action => 'remove_with_ajax' }
    format.html { if request.post?
      @item = @cart.remove params[:id]
      flash[:cart_notice] = "Eliminado <em>#{@item.game.name}</em>."
      redirect_to :controller => 'catalog'
    else
      render :controller => 'cart', :action => 'remove'
    end }
  end
end

def clear
  @page_title = 'Vaciar carrito'
  respond_to do |format|
    format.js { @cart.cart_items.destroy_all
      flash.now[:cart_notice] = "Carrito vaciado."
      render :controller => 'cart',
             :action => 'clear_with_ajax' }
    format.html { if request.post?
      @cart.cart_items.destroy_all
      flash[:cart_notice] = "Carrito vaciado."
      redirect_to :controller => 'catalog'
    else
      render :controller => 'cart', :action => 'clear'
    end }
  end
end

```

Fichero: app / views / catalog / games . html . erb

```
<dl id = 'games'>
  <% for game in @games %>
    <dt>
      <%= link_to game.name, :action => 'show', :id => game %>
      <%= link_to '+', :controller => 'cart', :action => 'add', :id => game, :remote => true %>
    </dt>
    <dd><%= game.developer.name %></dd>
    <dd><%= sprintf("Precio: %.2f €", game.price) %> </dd>
    <dd><small>Proveedores:</small>
      <% for supplier in game.suppliers %>
        <%= supplier.last_name %>, <%= supplier.first_name %>.
      <% end %>
    </small></dd>
    <% end %>
  </dl>
```

Fichero: app / views / cart / add_with_ajax . js . erb

```
jQuery.noConflict();
jQuery('#shopping_cart').html("<%= j render :partial => 'cart/cart' %>");
jQuery("#cart_item_<%= @item.game.id %>").css({'backgroundcolor':'#7ef'}).animate({'background-color':'#def'}, 3000);
jQuery('#cart_notice').hide('fade', {}, 3000);
```

Fichero: app / views / cart / clear_with_ajax . js . erb

```
jQuery.noConflict();
jQuery('#shopping_cart').html("<%= j render :partial => 'cart/cart' %>");
jQuery('#cart_notice').hide('fade', {}, 3000);
```

Fichero: app / views / cart / remove_with_ajax . js . erb

```
jQuery.noConflict();
jQuery('#shopping_cart').html("<%= j render :partial => 'cart/cart' %>");
jQuery('#cart_notice').hide('fade', {}, 3000);
```

Fichero: app / views / cart / cart . html . erb

```
<% if flash[:cart_notice] %>
<%= render :partial => 'cart/cart_notice' %>
<% end %>

<h3>Su carrito de la compra</h3>
<p>
  <strong>
    <% unless controller.controller_name == 'checkout' %>
      <%= link_to 'Proceder a la facturación', :controller => 'checkout' %>
    <% end %>
  </strong>
</p>
<ul>
  <% for item in @cart.cart_items %>
    <li id="cart_item_<%= item.game.id %>">
      <%= render :partial => 'cart/item', :object => item %>
    </li>
  <% end %>
</ul>
<p id='cart_total'><strong>Total: <%= sprintf "%0.2f €", @cart.total %></strong></p>
<% unless @cart.cart_items.empty? %>
  <p id='clear_cart_link'>
    <b><%= link_to 'Vaciar carrito', :controller => 'cart', :action => 'clear', :remote => true %></b>
  </p>
<% end %>
```

Fichero: app / views / cart / item . html . erb

```
<%= link_to item.game.name, :action => 'show', :controller => 'catalog', :id => item.game.id %>
<%= pluralize item.amount, "juego", "juegos" %>, <%= sprintf "%0.2f €", item.price * item.amount %>
(<%= link_to '<b>-</b>' .html_safe, :controller => 'cart', :action => 'remove', :id => item.game, :remote => true %>)
```

Fichero: app / assets / javascripts / application . js

```
// This is a manifest file that'll be compiled into application.js,
// which will include all the files listed below.
// ...
// It's not advisable to add code directly here, but if you do, it'll
// appear at the bottom of the compiled file.
// ...
//= require jquery
//= require jquery-ui/effects/effect-fade
//= require jquery-ui/effects/effect-highlight
//= require jquery_ujs
//= require turbolinks
//= require_tree .
```

5.9.6. Test

Modificamos los ficheros de los tests:

Fichero: test / integration / browsing_and_searching_test . rb

```
require 'test_helper'

class BrowsingTest < ActionDispatch::IntegrationTest
  fixtures :developers, :suppliers, :games, :games_suppliers

  test "browse" do
    jill = new_session_as :jill
    jill.index
    jill.second_page
    jill.game_details 'Pride and Prejudice'
    jill.latest_games
  end

  module BrowsingTestDSL
    include ERB::Util
    attr_writer :name
```

```

def index
  get '/catalog/index'
  assert_response :success
  assert_select 'dl#games' do
    assert_select 'dt', :count => 5
  end
  assert_select 'dt' do
    assert_select 'a', 'The Idiot'
  end
  check_game_links
end

def second_page
  get '/catalog/index?page=2'
  assert_response :success
  assert_template 'catalog/index'
  assert_equal Game.find_by_name('Pro Rails E-Commerce'),
               assigns(:games).last
  check_game_links
end

def game_details(name)
  @game = Game.where(:name => name).first
  get "/catalog/show/#{@game.id}"
  assert_response :success
  assert_template 'catalog/show'
  assert_select 'div#content' do
    assert_select 'h1', @game.name
    assert_select 'h2', "por #{@game.developer.name}"
  end
end

def latest_games
  get '/catalog/latest'
  assert_response :success
  assert_template 'catalog/latest'
  assert_select 'dl#games' do
    assert_select 'dt', :count => 5
  end
  @games = Game.latest(5)
  @games.each do |a|
    assert_select 'dt' do
      assert_select 'a', a.name
    end
  end
end

```



```
        end
    end
end
end

def rss
  get '/catalog/rss'
  assert_response :success
  assert_template 'catalog/rss'
  assert_match "application/xml", response.headers["Content-Type"]
  assert_select 'channel' do
    assert_select 'item', :count => 5
  end
  @games = Game.latest(5)
  @games.each do |game|
    assert_select 'item' do
      assert_select 'title', game.name
    end
  end
end

def check_game_links
  for game in assigns[:games]
    assert_select 'a' do
      assert_select '[href=?]', "/catalog/show/#{game.id}"
    end
  end
end

def new_session_as(name)
  open_session do |session|
    session.extend(BrowsingTestDSL)
    session.name = name
    yield session if block_given?
  end
end
end
```

Fichero: test / controllers / cart_controller_test . rb

```
require 'test_helper'

class CartControllerTest < ActionDispatch::IntegrationTest
  fixtures :developers, :suppliers, :games

  test "add" do
    assert_difference(CartItem, :count) do
      post '/cart/add', :params => { :id => 4 }
    end
    assert_response :redirect
    assert_redirected_to :controller => 'catalog'
    assert_equal 1, Cart.find(@request.session[:cart_id]).cart_items.size
  end

  test "remove" do
    post '/cart/add', :params => { :id => 4 }
    assert_equal [Game.find(4)], Cart.find(@request.session[:cart_id]).games

    post '/cart/remove', :params => { :id => 4 }
    assert_equal [], Cart.find(@request.session[:cart_id]).games
  end

  test "clear" do
    post '/cart/add', :params => { :id => 4 }
    assert_equal [Game.find(4)], Cart.find(@request.session[:cart_id]).games

    post '/cart/clear'
    assert_response :redirect
    assert_redirected_to :controller => 'catalog'
    assert_equal [], Cart.find(@request.session[:cart_id]).games
  end

  test "add_xhr" do
    assert_difference(CartItem, :count) do
      post '/cart/add', :params => { :id => 4 }, :xhr => true
    end
    assert_response :success
    assert_select_jquery :html, '#shopping_cart' do
      assert_select 'li#cart_item_4', /Pro Rails E-Commerce 4th Edition/
    end
    assert_equal 1, Cart.find(@request.session[:cart_id]).cart_items.size
  end

  test "remove_xhr" do
    post '/cart/add', :params => { :id => 4 }, :xhr => true
    assert_equal [Game.find(4)], Cart.find(@request.session[:cart_id]).games

    post '/cart/remove', :params => { :id => 4 }, :xhr => true
    assert_equal [], Cart.find(@request.session[:cart_id]).games
  end
```

```
test "clear_xhr" do
  post '/cart/add', :params => { :id => 4 }, :xhr => true
  post '/cart/add', :params => { :id => 1 }, :xhr => true
  assert_equal [Game.find(4), Game.find(1)], Cart.find(@request.session[:cart_id]).games

  post '/cart/clear', :params => { :id => 4 }, :xhr => true
  assert_response :success
  assert_equal [], Cart.find(@request.session[:cart_id]).games
end

end
```

Ejecutamos los dos tests:

```
ine@yii ~/projects/eshop1 $ rake test TEST=test/integration/browsing_and_searching_test.rb
ine@yii ~/projects/eshop1 $ rake test TEST=test/controllers/cart_controller_test.rb
```

5.9.7. Dificultades encontradas

Cuando pulsábamos “añadir”, se añadía dos veces. En el archivo Add_with_ajax corregimos un error de sintaxis.

5.9.8. Objetivos alcanzados

Gracias a las funcionalidades RSS y AJAX, ahora la tienda incorpora mensajes flash por lo que es más llamativa y atractiva para el usuario.

5.10. Sprint 10 (Borrado de órdenes)

Implementamos la función de borrar pedidos para el administrador de la tienda.

5.10.1. Sprint backlog y diagrama burndown

Tareas	L	M	X	J	V	S	D
Ficheros Migración	0	0	0	0	0	0	0
Vista y Controlador	30	0	0	0	0	0	0
Modelo	0	0	0	0	0	0	0
Test	0	0	0	0	0	0	0
Total	30	0	0	0	0	0	0

Sprint 10



5.10.2 Gemas utilizadas

No hemos añadido nuevas gemas.

5.10.3 Ficheros de migración

No ha sido necesario realizar ninguna migración.

5.10.4 Modelos ORM.

No procede en esta iteración.

5.10.5 Vistas y controladores

Implementamos el método destroy en el fichero:

Fichero: app / controllers / admin / order_controller . rb

```
class Admin::OrderController < Admin::AuthenticatedController
  def close
    order = Order.find(params[:id])
    order.close
    flash[:notice] = "El pedido ##{order.id} ha sido cerrado."
    redirect_to :action => 'index'
  end

  def show
    @order = Order.find(params[:id])
    @page_title = "Mostrando pedido ##{@order.id}"
  end

  def destroy
    @order = Order.find(params[:id])
    @order.destroy
    flash[:notice] = "Orden #{@order.id} eliminada correctamente."
    redirect_to :action => 'index'
  end

  def index
    @status = params[:id]
    if @status.blank?
      conditions = nil
      @status = 'all'
      @page_title = 'Listando todos los pedidos'
    else
      conditions = "status = '#{@status}'"
      @estado = 'abierto' if @status == 'open'
      @estado = 'procesado' if @status == 'processed'
      @estado = 'cerrado' if @status == 'closed'
      @estado = 'fallido' if @status == 'failed'
      @page_title = "Listando pedidos #{@estado.pluralize}"
    end
    @orders = Order.where(conditions).paginate(:page => params[:page], :per_page => 10
  ) end
end
```

Y modificamos el fichero de las vistas:

Fichero: app / views / admin / order / index . html . erb

```
<% if @orders == [] %>
<% if @status == 'all' %>
  <h2><%= "No hay pedidos." %></h2>
<% else %>
  <h2><%= "No hay pedidos con el estado '#{@estado}'." %></h2>
<% end %>
<% else %>
<table>
  <tr>
    <th>ID</th>
    <th>Estado</th>
    <th>Precio total</th>
    <th>Cantidad</th>
    <th>Creado el</th>
    <th>Actualizado el</th>
    <th></th>
  </tr>
<% for order in @orders %>
  <tr>
    <td align="center"><%= order.id %></td>
    <% estado = 'abierto' if order.status == 'open' %>
    <% estado = 'fallido' if order.status == 'failed' %>
    <% estado = 'procesado' if order.status == 'processed' %>
    <% estado = 'cerrado' if order.status == 'closed' %>
    <% order.status[0].capitalize + order.status[1..order.status.length-1] %>
    <td align="center"><%= estado.capitalize %></td>
    <td align="center"><%= order.total %></td>
    <td align="center"><%= order.amount %></td>
    <td align="center"><%= order.created_at.strftime("%d-%m-%Y %I:%M") %></td>
    <td align="center"><%= order.updated_at.strftime("%d-%m-%Y %I:%M") %></td>
    <td><%= link_to 'Mostrar', :action => 'show', :id => order %></td>
    <td><%= button_to 'Borrar', { :action => 'destroy', :id => order },data: { confirm:
      "¿Seguro que quieres eliminar la orden #{order.id}?" } %></td>    </tr>
<% end %>
</table>

<% if @orders.total_pages > 1 %>
  <br/>
  <%= 'Página:' %>
<% end %>

<%= will_paginate @orders, :page_links => true, :link_separator => ' ', :container => false,
                     :previous_label => '', :next_label => '' %>
<p></p>
<% end %>

<%= render :partial => 'navigation' %>
```

5.10.6 Test

No se han hecho nuevos tests en esta iteración.

5.10.7 Dificultades encontradas

Esta implementación ha sido bastante sencilla como hemos podido observar en los cambios a realizar en los ficheros, no hemos encontrado ninguna dificultad.

5.10.8 Objetivos alcanzados

Ahora el administrador de la tienda puede borrar las órdenes cuando lo estime necesario.

6. Componentes del grupo

Jesús Facio Treceño

Coordinador

Desarrollador de apoyo

Firma: _____

Miguel Afán Espinosa

Desarrollador primario

Supervisor

Firma: _____

Carmen Ruiz de Celis

Supervisión de tests

Desarrollo de documentación

Firma: _____

Claudia Soriano Roldán

Administración de bases de datos

Desarrollo de la documentación

Firma: _____

7. Conclusiones

Gracias a este proyecto nos hemos familiarizado e iniciado en Ruby y Ruby on Rails y además hemos aprendido a desarrollar de manera efectiva, fácil, guiada y sólida una aplicación funcional de comercio electrónico.

Todos los miembros del equipo coincidimos en que este aprendizaje nos será de ayuda en el futuro y en el mundo laboral.

