Item 10, A+ - Implementando los pagos de Acmeshop a través de Paypal

Acmeshop es un mercado online, y como tal, la base del proyecto debe ser un sistema de pagos entre usuarios rápido, cómodo y fiable. Implementar nuestra propia solución conllevaría mucho tiempo y problemas, así que decidimos utilizar la plataforma de pagos online Paypal, principalmente por las ventajas expuestas en el documento del Item 9.

Paypal nos permite implementar pagos en nuestra aplicación con un desarrollo relativamente sencillo, ya que su API REST nos permite hacer llamadas y redirigir al usuario hacia su sistema, que se encarga de la autenticación y asegura la seguridad del dinero de los usuarios. En nuestra implementación, esto se hace a través de su librería javascript, *checkout.js*. Además, nos ayudamos del *Paypal SDK* para Java, que nos ahorra el trabajo de crear POJOs que representen las diferentes peticiones y respuestas de la API, así como realizar las llamadas a dicha API y controlar errores.

A continuación definimos el flujo a través del cual gestionamos un pago de un usuario:

1. El usuario accede al formulario de pagos. A este formulario se puede acceder desde cuatro sitios diferentes:
    a. Para pagar su carro de la compra: El carro de la compra de un usuario está almacenado en la base de datos, por lo que el controlador sólo calcula el total a pagar y lo muestra.
    b. Para pagar el precio instantáneo de una subasta: El controlador guarda en sesión el id de la subasta que quiere pagar el usuario, para poder recuperarla más adelante.
    c. Para pagar una subasta ganada: igual que el caso anterior.
    d. Para pagar el servicio Premium: El controlador simplemente muestra el precio del servicio. Este formulario es el único que no tiene ningún campo para datos de envío y facturación, sólo incluye el botón de pago y el de cancelación.

    (Nótese que evitamos pasar a través del formulario ningún dato que se vaya a utilizar a la hora de realizar el pago, para evitar posibles actividades maliciosas.)

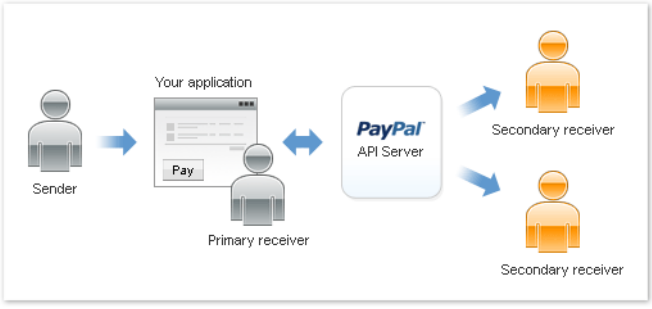
2. En el formulario, la librería *checkout.js* carga el botón de Paypal. Cuando el usuario rellena el formulario y pulsa dicho botón, serializamos el formulario y mandamos una petición AJAX a nuestro controlador con los datos de dicho formulario.
3. El controlador valida el formulario. Si no hay errores, crea una petición y la envía a la API de Paypal para crear un objeto *Payment*. Este objeto contiene los detalles del pago, excepto los del pagador, que deberá rellenar él mismo en el siguiente paso[1].
4. Si el objeto se crea correctamente, el controlador devuelve el ID de dicho pago. La librería *checkout.js*, con ese ID, abre una ventana que conecta al usuario con el sistema de Paypal. El usuario inicia sesión (también puede pagar directamente con una tarjeta) y acepta el pago. Puede pagar mediante tarjeta, con una cuenta bancaria asociada o su saldo de Paypal.

[1] Actualmente Paypal no soporta todos los detalles que nos gustaría introducir en el objeto Payment, como por ejemplo el desglose de artículos, en su nueva API REST.

5. Se realiza una petición AJAX a otro controlador, que recupera el objeto *Payment* mediante una petición GET a la API de Paypal. Para ejecutar el pago, envía una nueva petición a paypal, que incluye dicho objeto. Si Paypal da una respuesta afirmativa, el pago ha sido ejecutado correctamente y el dinero se moverá de la cuenta del usuario a la cuenta central de Acmeshop. El controlador puede proceder a ejecutar la lógica de confirmación de la compra: Creación del ticket, actualización de los stocks, aplicar status de Premium al usuario…
6. (Excepto en las compras del servicio Premium) Ahora Acmeshop debe reenviar este pago a los vendedores implicados. Puede que la compra sea de un comprador a un solo vendedor, o a varios vendedores en el caso de estar pagando su carro de la compra. Para acomodar ambos casos, utilizamos la API de *Payouts* de Paypal. Un *Payout* es el método que ofrece Paypal para realizar un pago a varios destinatarios al mismo tiempo. Creamos un objeto *Payout*, compuesto de tantos Ítems como vendedores, y lo ejecutamos. Cada ítem tiene un destinatario (dirección de correo electrónico[2]), la cantidad que le corresponde (menos posibles comisiones e impuestos) y un ID que identifica ese pago en caso de que se produzca alguna reclamación.
7. Si todo ha funcionado correctamente, se devuelve una ventana de confirmación. El pago ha finalizado.

En general, Paypal es un servicio muy útil con una API que tiene muchos años de desarrollo detrás. Sin embargo, nos hemos encontrados con algunas dificultades durante el desarrollo:

- La documentación de Paypal no es fácil de navegar. Los documentos de la antigua API SOAP se mezclan con los de la nueva REST, es difícil encontrar lo que buscamos a través de muchos menús con nombres parecidos.
- La antigua API SOAP, que tiene la funcionalidad completa de Paypal, está deprecada, y muchas de sus funciones no están disponibles para nuevas aplicaciones. En concreto, la API de *Adaptative Payments* contiene un tipo de pago, *Chained Payments*, que resuelve por completo nuestro caso de uso sin tener que dividirlo en dos tipos de pago diferentes (*Payment* del comprador a la aplicación y *Payout* de la aplicación a los vendedores), contemplando incluso las comisiones e impuestos que quisiéramos cobrar:



1 Descripción gráfica de un Chained Payment

[2] En Paypal, todos los pagos se realizan hacia una dirección de correo electrónico. Si el destinatario no tiene cuenta de Paypal, le llegará un correo electrónico solicitando que se haga una cuenta para poder recibir el pago.

- La que más nos ha dificultado el desarrollo, es el mal funcionamiento de los servidores *sandbox* de Paypal. Paypal permite a las aplicaciones probar sus integraciones mediante un entorno *sandbox* en el cual se utilizan cuentas y dinero ficticio para poder realizar transacciones de prueba sin dinero real. Lo malo de este entorno es que los servidores asignados no funcionan muy bien, lo cual produce que las transacciones tarden mucho en reflejarse (desde unos minutos hasta más de un día) y que a veces, durante períodos de tiempo que llegan a ser de hasta horas, el servidor devuelva errores 500 a cualquier petición. Por suerte, esto sólo ocurre en el entorno *sandbox*.

Cómo probar la integración en Acmeshop:

La forma más fácil de probar la integración con Acmeshop es iniciar sesión como "user1", "user1"; acceder a cualquier Auction Advertisement y pagar el precio de compra instantánea (buy now). Esto nos lleva directamente al formulario de pago. Dejar los datos que se rellenan automáticamente y pulsar el botón de paypal, iniciar sesión en paypal con la cuenta "user1@acmeshop.com", "acmeshop" y confirmar el pago.

Para comprobar que el pago se realiza correctamente, se puede acceder al historial de la cuenta de Paypal del usuario, de la aplicación o del vendedor. Por desgracia, como hemos dicho antes, el pago puede tardar más de un día en aparecer allí, así que la consola del servidor muestra las respuestas que ha devuelto la API de Paypal. Puede comprobarse que tanto el Payment como el Payout se han ejecutado correctamente.

Las cuentas *sandbox* que hemos configurado para hacer pruebas en la aplicación son:

- Cuenta de la aplicación, recibe los pagos de los compradores y los reenvía a los vendedores: **"payments@acmeshop.com"**
- Cuentas de User:
  - **"user1@acmeshop.com"**
  - **"user2@acmeshop.com"**
  - **"user3@acmeshop.com"**
- Cuentas de Business:
  - **"business1@acmeshop.com"**
  - **"business2@acmeshop.com"**
  - **"business3@acmeshop.com"**

La contraseña para todas las cuentas es **acmeshop**, y para ver sus historiales de actividad se debe iniciar sesión en **sandbox.paypal.com**