A+ - PayPal

PayPal es un sistema de pagos en línea que soporta transferencias de dinero entre usuarios y sirve como una alternativa electrónica a los métodos de pago tradicionales además de ofrecer seguridad en los pagos a través de Internet. Es por ello por lo que hemos decidido incorporar PayPal como una forma de pago alternativa al uso de la tarjeta de crédito.

Suponemos que se ha creado una cuenta en la [web de desarrolladores de Paypal](https://developer.paypal.com/) y que ha obtenido su clave pública y privada, en nuestro caso estas serán:

AW1jnKlMWtcJ89S06Cb\_3wvUC2EezkhFwKCm0oJWov6wExVxI1q6rKr1My3Hafc6s41rJc-yx-etXV1q

EA68rBYMMp1OYW9Oc\_IF4TIBC7\_AND2M6j9baEb1-ijh7l\_8qgXgZ1moX\_rJjX2SKAa4wGsyCG3fp86\_

Primero debemos añadir como dependencia el SDK de PayPal en nuestro pom.xml para realizar la integración de PayPal de forma más sencilla que utilizando otros métodos.

*<!-- Paypal Dependency -->*

        <dependency>

            <groupId>com.paypal.sdk</groupId>

            <artifactId>rest-api-sdk</artifactId>

            <version>1.13.1</version>

        </dependency>

Ahora creamos un nuevo *package* dentro de *services*, donde creamos la clase **PaypalClient.java** en la que haciendo uso del SDK de PayPal escribiremos el código necesario para que actúe de cliente para la API de PayPal. Primeramente, creamos un método llamado ***createPayment*** al que le pasaremos el precio a pagar, el identificador de la rifa, la cantidad de productos que queremos comprar y el objeto request que se crea por cada petición HTTP a un servlet y nos devolverá un mapa con objetos donde vendrá, entre otras cosas, la URL donde el usuario iniciará sesión.

public class PaypalClient {

*/\**

*\* Guardamos como propiedades privadas de la clase el clientId y clientSecret.*

*\*/*

    private String clientId = "AW1jnKlMWtcJ89S06Cb\_3wvUC2EezkhFwKCm0oJWov6wExVxI1q6rKr1My3Hafc6s41rJc-yx-etXV1q";

    private String clientSecret = "EA68rBYMMp1OYW9Oc\_IF4TIBC7\_AND2M6j9baEb1-ijh7l\_8qgXgZ1moX\_rJjX2SKAa4wGsyCG3fp86\_";

    public Map<String, Object> createPayment(final String sum, final int raffleId, final int amountItem, final HttpServletRequest request){

     Map<String, Object> response;

     Amount amount;

     Transaction transaction;

     List<Transaction> transactions;

     Payer payer;

     Payment payment;

     String scheme, serverName, contextPath;

     int portNumber;

     RedirectUrls redirectUrls;

     Payment createdPayment;

     String redirectUrl;

     APIContext context;

     List<Links> links;

*/\**

*\* Creamos un mapa que servirá como respuesta donde*

*\* irá la información necesaria para completar el pago*

*\*/*

     response = *new* HashMap<String, Object>();

*/\**

*\* Primero de todo, debemos indica la moneda en la que*

*\* se realizará la transacción e indicaremos la cantidad.*

*\* Dicha cantidad corresponderá con el precio del tique*

*\* multiplicado por la cantidad que ha comprado del mismo.*

*\*/*

     amount = *new* Amount();

     amount.setCurrency("EUR");

     amount.setTotal(sum);

*/\**

*\* Comenzamos la transacción monetaria poniéndole*

*\* el objeto amount que hemos creado previamente.*

*\*/*

     transaction = *new* Transaction();

     transaction.setAmount(amount);

*/\**

*\* La transacción debe ir en una lista de transacciones*

*\* por si quiere añadir más de una. En nuestro caso*

*\* no será necesario porque hemos decido que aunque*

*\* compre más de un tique, el pago será único.*

*\*/*

     transactions = *new* ArrayList<Transaction>();

     transactions.add(transaction);

*/\**

*\* Creamos un objeto de la clase Payer y*

*\* marcamos como método de pago paypal,*

*\* otros modos de pago podrían ser a través*

*\* de tarjeta de crédito (credit\_card)*

*\* o cuenta bancaria (bank) entre otros*

*\*/*

     payer = *new* Payer();

     payer.setPaymentMethod("paypal");

*/\**

*\* Creamos el pago, en el cual indicaremos*

*\* con el método setIntent que será una venta*

*\* y que el pago se hará a través de PayPal*

*\* (utilizando payer) y por último le pasamos*

*\* el listado de transacciones con setTransactions.*

*\*/*

     payment = *new* Payment();

     payment.setIntent("sale");

     payment.setPayer(payer);

     payment.setTransactions(transactions);

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

scheme = request.getScheme();

serverName = request.getServerName();

portNumber = request.getServerPort();

contextPath = request.getContextPath();

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*

*/\**

*\* Indicamos las URLs a las que se redigirá cuando*

*\* se cancele la compra (setCancelUrl) o cuando termine*

*\* (setReturnUrl)*

*\*/*

     redirectUrls = *new* RedirectUrls();

     redirectUrls.setCancelUrl(scheme+"://"+serverName+":"+portNumber+contextPath+"/raffle/display.do?raffleId="+raffleId);

     redirectUrls.setReturnUrl(scheme+"://"+serverName+":"+portNumber+contextPath+"/ticket/user/completepayment.do?raffleId="+raffleId+"&amount="+amountItem);

     payment.setRedirectUrls(redirectUrls);

*try* {

     redirectUrl = "";

*/\**

*\* Creamos el contexto de la aplicación con el*

*\* que se nos generará una URL que dirigirá al*

*\* usuario a una vista donde poder realizar*

*\* el pago autenticándose en PayPal.*

*\*/*

     context = *new* APIContext(clientId, clientSecret, "sandbox");

     createdPayment = payment.create(context);

*if*(createdPayment!=null){

     links = createdPayment.getLinks();

*for* (Links link*:*links) {

*if*(link.getRel().equals("approval\_url")){

     redirectUrl = link.getHref();

*break*;

     }

     }

     response.put("status", "success");

     response.put("redirect\_url", redirectUrl);

     }

     } *catch* (PayPalRESTException e) {

*throw* *new* IllegalArgumentException("Error happened during payment creation!");

     }

*return* response;

    }

Una vez escrito el método, en el controlador rescatamos la URL de la que hablamos antes y redirigimos con el ModelAndView a dicha dirección y a su vez le enviamos como objetos el resto de los objetos que nos devolvió el método ***createPayments***.

attributes = *this*.paypalClient.createPayment(String.valueOf(raffle.getPrice()\*amount), raffleId, amount, request);

result = *new* ModelAndView("redirect:"+(String) attributes.get("redirect\_url"));

*for*(String key*:* attributes.keySet())

result.addObject(key, attributes.get(key));

Tras ser redirigido el usuario a la web de PayPal y una vez introducida todas las credenciales se redirige a otra dirección del controlador con dos identificadores: el de pago (payment) y el del pagador (payer). Dichos valores se le pasarán al método que crearemos ahora: ***completePayment*** para que se complete el pago.

public Map<String, Object> completePayment(final String paymentId, final String PayerId){

     Map<String, Object> response;

     Payment payment;

     PaymentExecution paymentExecution;

     APIContext context;

     Payment createdPayment;

*/\**

*\* Creamos un mapa que devolveremos con los objetos*

*\* creados en este método.*

*\*/*

     response = *new* HashMap<String, Object>();

*/\**

*\* Creamos un objeto de la clase payment*

*\* y le ponemos el identificador de pago*

*\* que nos generó PayPal al terminar en la*

*\* vista de pago.*

*\*/*

     payment = *new* Payment();

     payment.setId(paymentId);

*/\**

*\* Realizamos lo mismo pero esta vez con el*

*\* identificador del pagador.*

*\*/*

     paymentExecution = *new* PaymentExecution();

     paymentExecution.setPayerId(PayerId);

*try* {

     context = *new* APIContext(clientId, clientSecret, "sandbox");

*/\**

*\* Ejecutamos el pago para hacerlo efectivo y*

*\* que se descuente el dinero en la cuenta de*

*\* PayPal del usuario.*

*\*/*

     createdPayment = payment.execute(context, paymentExecution);

*if*(createdPayment!=null){

     response.put("status", "success");

     response.put("payment", createdPayment);

     }

     } *catch* (PayPalRESTException e) {

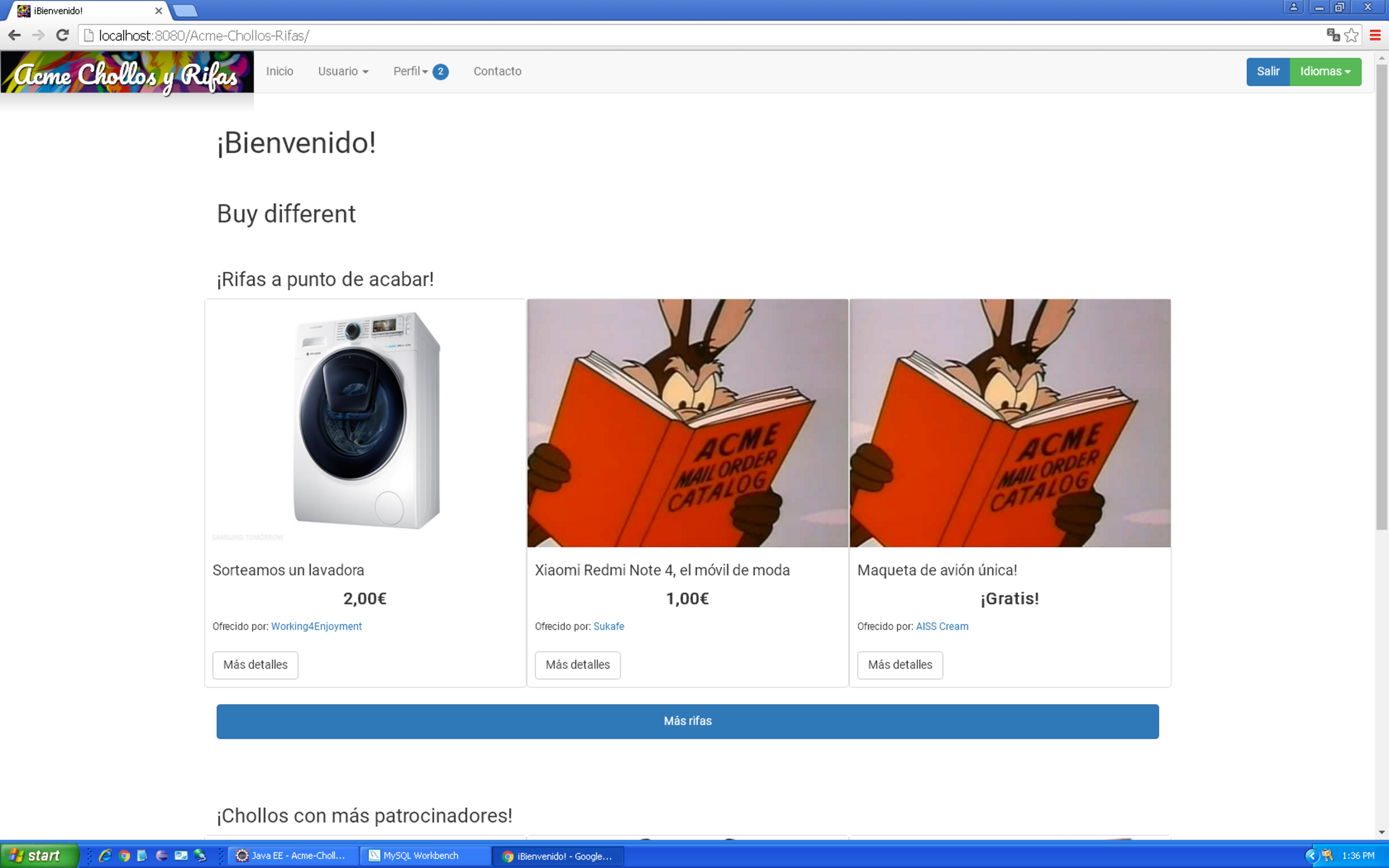
*throw* *new* IllegalArgumentException(e);

     }

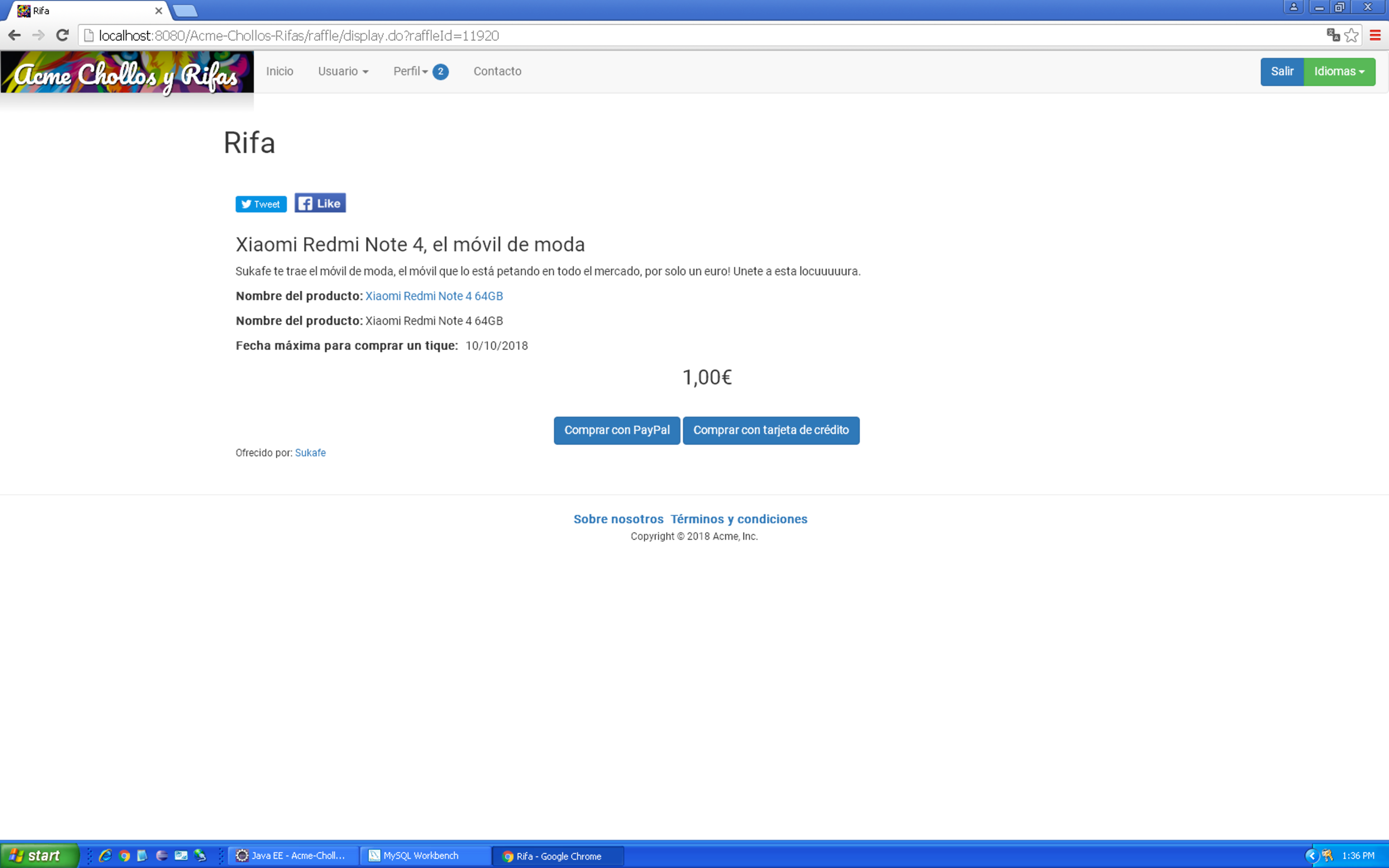
*return* response;

    }

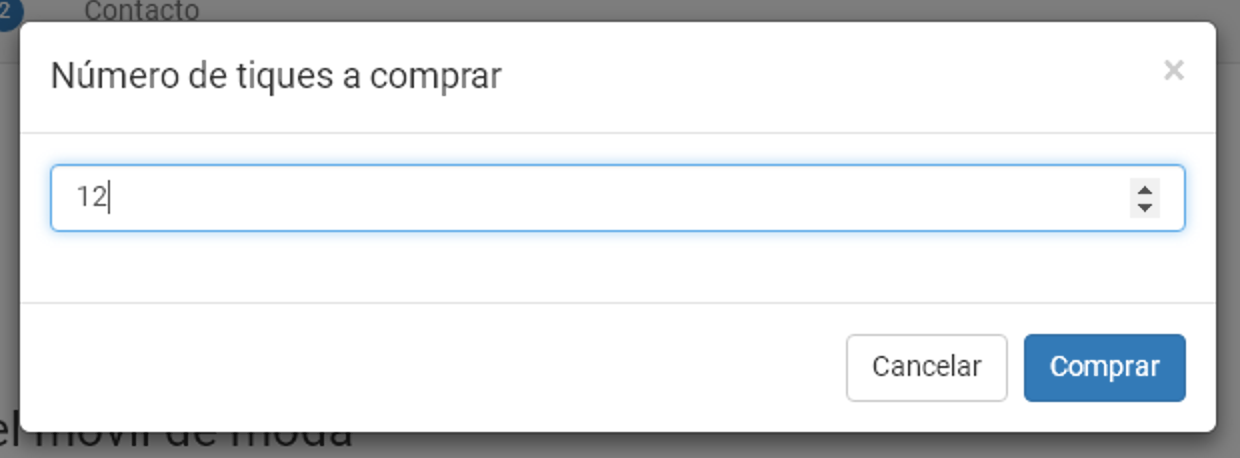
Una vez completado el pago se redirigirá a la lista de tiques donde podemos comprobar que se ha realizado la compra de forma efectiva. Para comprobar que todo está correcto vamos a proceder a realizar un ejemplo:



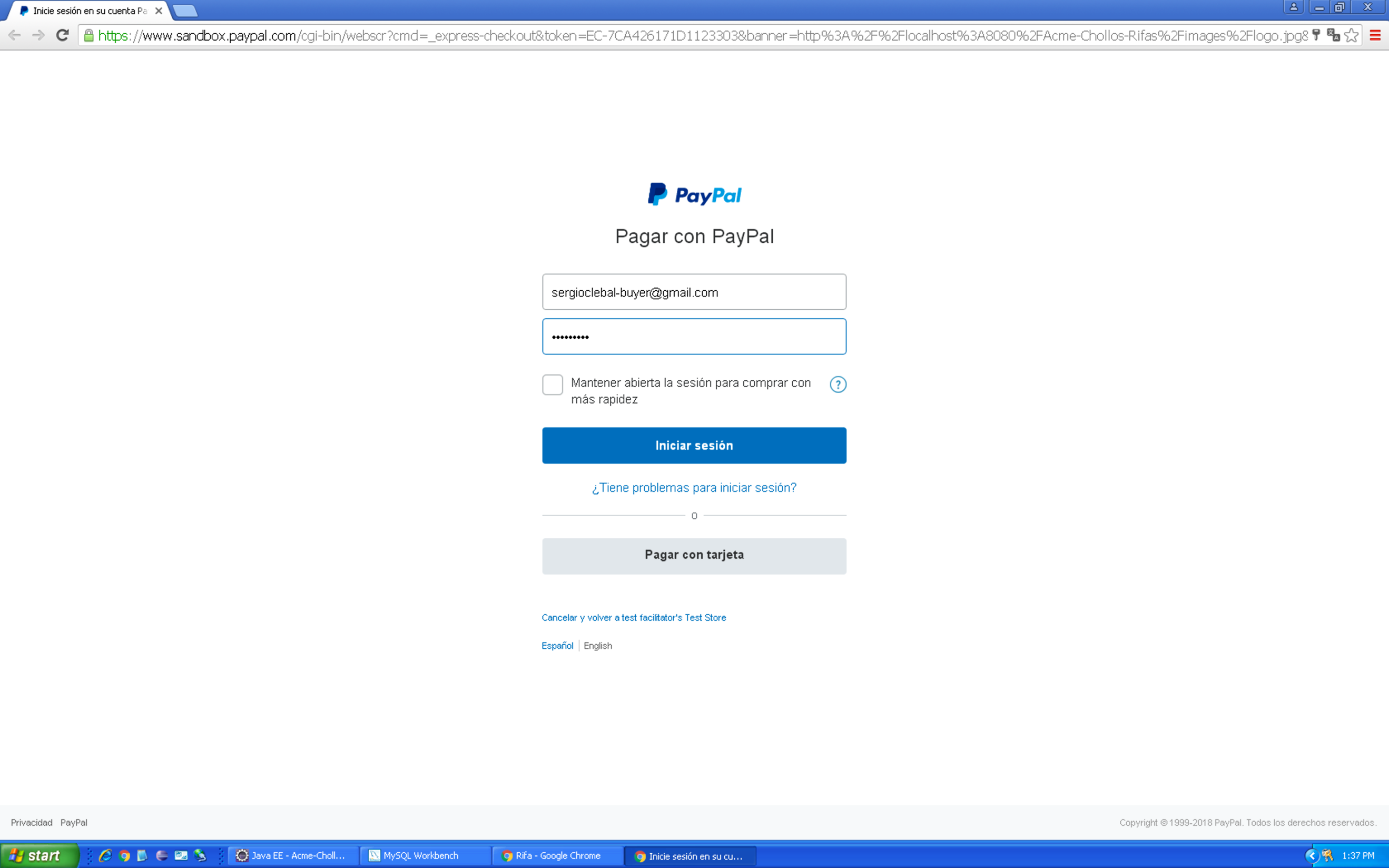
Comenzamos en la web autenticados como usuario, por ejemplo, utilizando las credenciales (user1 user1). Entramos en la vista de despliegue de la rifa “Xiaomi Redmi Nota 4, el móvil de moda”:



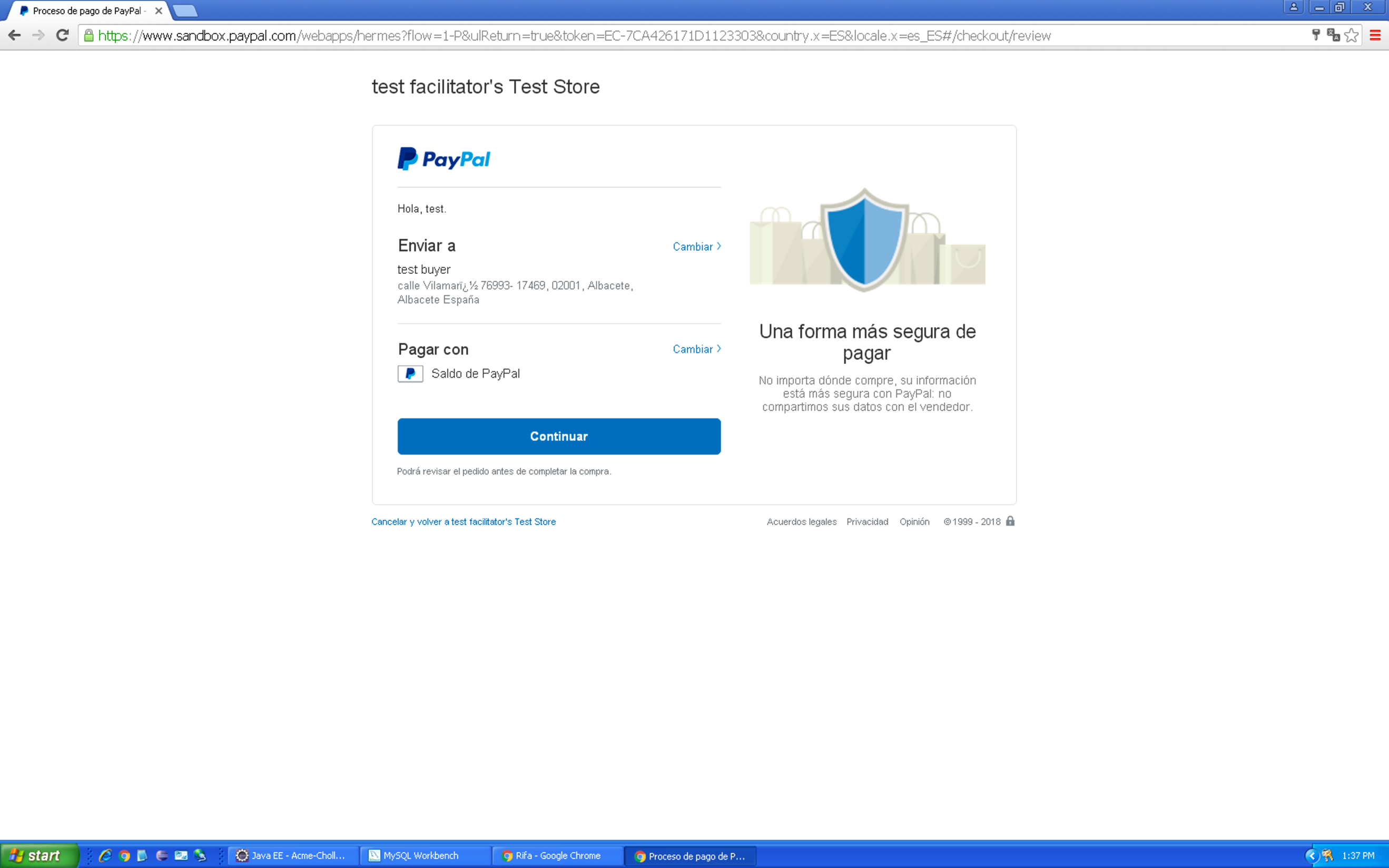
Y nos dará la opción de comprar tiques utilizando PayPal o una tarjeta de crédito, en nuestro caso seleccionamos PayPal para probarlo. Entonces nos mostrará un modal en el que indicaremos la cantidad que vamos a comprar, en este caso pondremos una cantidad notable, 12, por ejemplo:



Una vez rellenado dicho formulario se llamará al método createPayment que nos generará la URL a la que seremos redirigidos donde veremos la siguiente pantalla para iniciar sesión en PayPal, en este caso utilizaremos una cuenta falsa con los credenciales ([sergioclebal-buyer@gmail.com](mailto:sergioclebal-buyer@gmail.com) acas41997):



Ya iniciada la sesión utilizamos el saldo de PayPal para pagar los tiques y pulsamos en continuar:



Si todo ha salido bien seremos redirigidos al listado de tiques donde navegaremos a la última página y ahí podemos comprobar que efectivamente se ha realizado la compra de los 12 tiques, por lo que podemos concluir el resultado ha sido satisfactorio.

