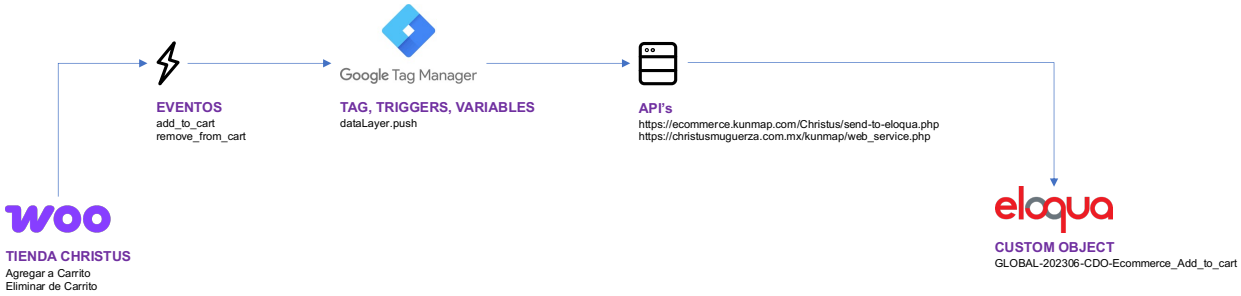
DOCUMENTO
INTEGRACIÓN ADD TO CART ECOMMERCE

Resumen Rápido

Este script corre dentro de un tag de Google Tag Manager (Custom HTML) cuando se dispara el evento "add to cart". Lee/actualiza un contador por sesión, construye un `payload` con variables de GTM, y hace un **POST JSON** a un API intermedia (`https://ecommerce.kunmap.com/Christus/send-to-eloqua.php`). Esa API procesa, transforma y envía la información a Eloqua.

Arquitectura



Endpoints

- `https://ecommerce.kunmap.com/Christus/send-to-eloqua.php`
- `https://christusmuguerza.com.mx/kunmap/web_service.php`

1) Tag Add to Cart Script GTM (versión con comentarios)

```
<script>
(function() { // IIFE: función que se ejecuta inmediatamente para aislar variables del scope
global
  // Obtener el contador actual de sessionStorage (clave 'cartProductCount'), o empezar en 0
  var productCount = sessionStorage.getItem('cartProductCount');
  productCount = productCount ? parseInt(productCount) : 0;

  // Limitar a 3 letras: a (0), b (1), c (2) → después de c vuelve a a
  // 97 = código ASCII de 'a'
  var indexLetter = String.fromCharCode(97 + (productCount % 3));

  // Guardar el nuevo valor del contador para la próxima vez
  sessionStorage.setItem('cartProductCount', productCount + 1);

  // Espera breve antes de enviar (150ms)
  setTimeout(function() {
    var payload = {
      email: "{{Correo cliente}}",
      service_name: "{{Nombre servicio}}",
      sku: "{{SKU servicio}}",
      category: "{{Categoria servicio}}",
      quantity: "{{Cantidad servicio}}",
      price: "{{Precio servicio}}",
      image_url: "{{URL Imagen servicio}}",
      timestamp: "{{Timestamp}}",
      medical_unit: "{{Unidad medica}}",
      url: "{{URL Servicio}}",
      order: indexLetter,
      pricem: "{{Precio member}}",
      cita: "{{Cita servicio}}",
      discount: "{{Descuento Servicio}}",
      utmSource: "{{utm_source}}",
      utmCampaign: "{{utm_campaign}}",
      utmMedium: "{{utm_medium}}"
    };

    console.log("Sending payload:", payload);

    fetch("https://ecommerce.kunmap.com/Christus/send-to-eloqua.php", {
      method: "POST",
      headers: {
        "Content-Type": "application/json"
      },
      body: JSON.stringify(payload)
    }).then(function(response) {
      if (!response.ok) {
        console.error("Webhook error:", response.status, response.statusText);
      } else {
        console.log("Data sent to Eloqua webhook.");
      }
    }).catch(function(error) {
        console.error("Fetch error:", error);
      });
  }, 150); // delay de 150ms
})();
</script>
```

2) Explicación detallada de cada función/bloque

```
(function() {
```

- Crea un scope local para variables y evita contaminar el window global. Se ejecuta inmediatamente al cargarse el tag.

```
sessionStorage.setItem('cartProductCount', productCount + 1);
```

- `sessionStorage` guarda datos por pestaña/navegación: persiste mientras la pestaña esté abierta.
- `getItem('cartProductCount')` recupera el contador de productos añadidos en la sesión actual.
- `setItem('cartProductCount', productCount + 1)` incrementa el contador para el siguiente producto.
- Nota: `sessionStorage` es por origen (domain + protocolo + puerto) y se borra al cerrar la pestaña.

```
productCount = productCount ? parseInt(productCount) : 0;
```

- Si existe `productCount`, se convierte a entero; si no existe, se usa `0`. Evita `NaN`.

```
var indexLetter = String.fromCharCode(97 + (productCount % 3));
```

- Convierte un número ASCII a letra. 97 = 'a'.
- `productCount % 3` produce 0,1,2 → resultando en 'a','b','c' cíclicamente.
- Ejemplo paso a paso: si `productCount = 5`, `5 % 3 = 2`, `97 + 2 = 99`, `String.fromCharCode(99) = 'c'`.

```
setTimeout(function() {
    …
}, 150); // delay de 150ms
```

- Retrasa la ejecución del bloque de envío 150 ms. Espera a que GTM resuelva variables o que el dataLayer esté totalmente poblado.

```
var payload = {
      email: "{{Correo cliente}}",
      service_name: "{{Nombre servicio}}",
      sku: "{{SKU servicio}}",
      category: "{{Categoria servicio}}",
      quantity: "{{Cantidad servicio}}",
      price: "{{Precio servicio}}",
      image_url: "{{URL Imagen servicio}}",
      timestamp: "{{Timestamp}}",
      medical_unit: "{{Unidad medica}}",
      url: "{{URL Servicio}}",
      order: indexLetter,
      pricem: "{{Precio member}}",
      cita: "{{Cita servicio}}",
      discount: "{{Descuento Servicio}}",
      utmSource: "{{utm_source}}",
      utmCampaign: "{{utm_campaign}}",
      utmMedium: "{{utm_medium}}"
    };
```

- Objeto con campos que el API intermedia espera. Cada valor `{{...}}` es una variable de GTM (Data Layer Variable o Variable de Primera Parte).
- Importante validar tipos: `quantity` debería ser número, `price/pricem/discount` números (floats), `timestamp` en ISO8601 preferiblemente.

```
    console.log("Sending payload:", payload);
```

- Útil para debugging en consola del navegador. Muestra el objeto antes de enviarlo.

```
fetch("https://ecommerce.kunmap.com/Christus/send-to-eloqua.php", {
      method: "POST",
      headers: {
        "Content-Type": "application/json"
      },
      body: JSON.stringify(payload)
    }).then(function(response) {
      if (!response.ok) {
        console.error("Webhook error:", response.status, response.statusText);
      } else {
        console.log("Data sent to Eloqua webhook.");
      }
    }).catch(function(error) {
        console.error("Fetch error:", error);
      });
},
```

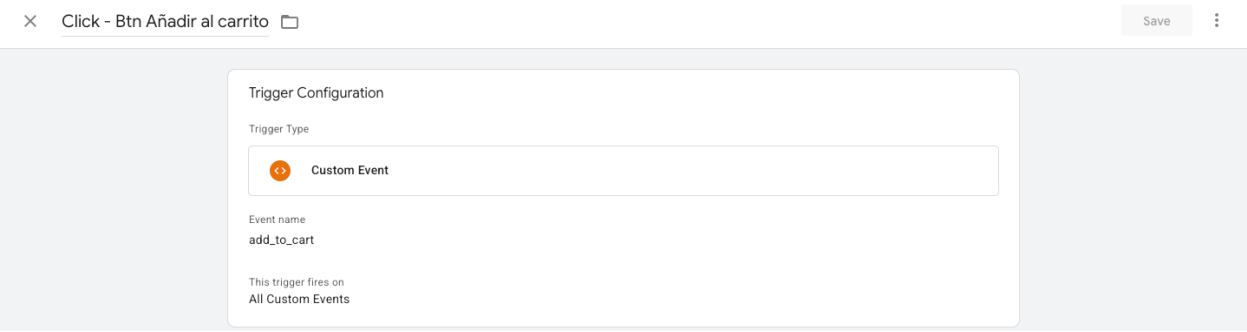
- Envía un `POST` con `Content-Type: application/json` y cuerpo serializado.
- `.then(response => ...)` comprueba si `response.ok` (status 200–299) — si no, registra error con status.
- `.catch(...)` captura errores de red (p. ej. CORS bloqueado, no disponibilidad del endpoint).

3) Ecommerce DataLayer Array

```
ecommerce: {
    currency: "MXN",
    value: 1000,
    items: [
        {
            item_id: 53120302830,
            item_name: "Anticipo paquete de maternidad – parto natural",
            sku: "CMAE354840",
            price: 1000,
            stocklevel: null,
            stockstatus: "instock",
            google_business_vertical: "retail",
            item_category: "Uncategorized",
            id: 53120302830,
            unidad: "CHRISTUS MUGUERZA Hospital Alta Especialidad",
            especialidad: "",
            estudio: "Paquetes de maternidad",
            product_url:    "https://christusmuguerza.com.mx/producto/antici" +
                            "po-paquete-de-maternidad-parto-natural-hospital" +
                            "-alta-especialidad/",
            product_image_url:  "https://christusmuguerza. com.mx/wp-conten" +
                                "t/uploads/2023/10/maternidad.jpg",
            datetime: "2025-07-04T22:53:53+00:00"
        }
    ]
},
```

4) Trigger Custom Event

En GTM, un **trigger** (o disparador) es una condición que, cuando se cumple, hace que un tag se ejecute. En este caso, queremos que el trigger se active **únicamente** cuando se produce un evento de “agregar al carrito” o “add_to_cart”. El evento está configurado como Custom Event.



5) Variables

Nombre	Tipo	DataLayer Name
Cantidad servicio	Data Layer Variable	cartContent.items.0.quantity
Categoria servicio	Data Layer Variable	ecommerce.items.0.estudio
Cita servicio	Data Layer Variable	ecommerce.detail.agenda
Correo cliente	Data Layer Variable	visitorEmail

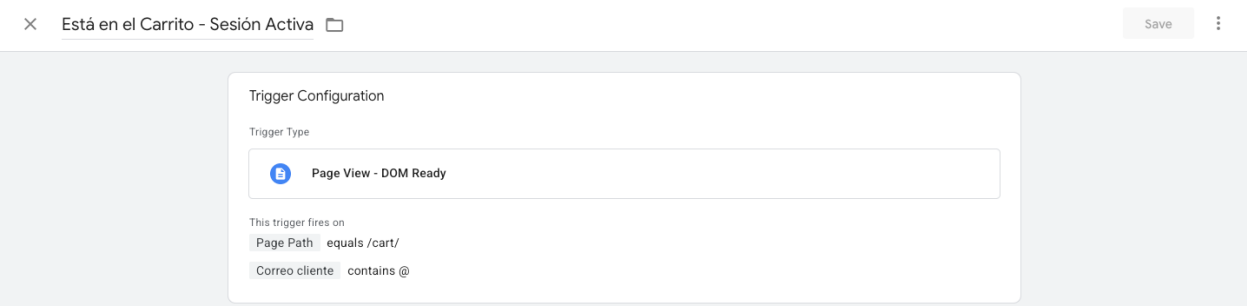
Descuento Servicio	Data Layer Variable	ecommerce.detail.discount_type
Nombre servicio	Data Layer Variable	ecommerce.items.0.item_name
Num Cart	Data Layer Variable	ecommerce.detail.order
Precio member	Data Layer Variable	ecommerce.detail.discount_price
Precio servicio	Data Layer Variable	ecommerce.items.0.price
Productos relacionados	Data Layer Variable	ecommerce.detail.related_products
SKU servicio	Data Layer Variable	ecommerce.items.0.sku
SKU eliminado	Data Layer Variable	ecommerce.detail.remove_id
Timestamp	Data Layer Variable	ecommerce.items.0.datetime
Unidad medica	Data Layer Variable	ecommerce.items.0.unidad
URL Imagen servicio	Data Layer Variable	ecommerce.items.0.product_image_url
URL Servicio	Data Layer Variable	ecommerce.items.0.product_url
utm_campaign	Data Layer Variable	utm_campaign
utm_medium	Data Layer Variable	utm_medium
utm_source	Data Layer Variable	utm_source

Tag Página de Carrito - Eventos Personalizados Script GTM

```
<script>
    document.addEventListener("click", function (event) {
        if (event.target && event.target.classList.contains("remove")) {

            var button = event.target;
            var productId = button.getAttribute("data-product_sku");

            dataLayer.push({
                event: "eliminar_carrito",
                "ecommerce.detail.remove_id": productId,
            });

        }
    });
</script>
```

Este script está diseñado para **escuchar clics** en elementos de la página y, cuando el usuario hace clic en un botón con la clase `remove`, **enviar un evento al Data Layer** de Google Tag Manager (GTM) con información del producto eliminado.

Trigger Page View DOM Ready



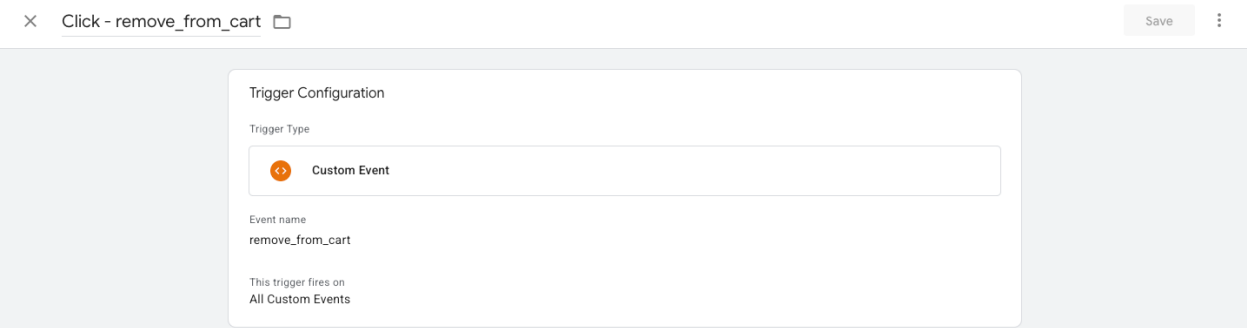
Tag Remove from cart Script GTM

```
<script>
 var xhr = new XMLHttpRequest();
xhr.open("POST", "https://christusmuguerza.com.mx/kunmap/web_service.php", true);
xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
xhr.onreadystatechange = function () {
    if (xhr.readyState === 4 && xhr.status === 200) {
        var respuesta = xhr.responseText;
        console.log(respuesta);
    }
};

var datos = "email=" + encodeURIComponent({{Correo cliente}}) +
    "&sku=" + encodeURIComponent({{SKU eliminado}}) +
    "&type=" + encodeURIComponent('remove');
xhr.send(datos);

</script>
```

Este script, configurado como una etiqueta en Google Tag Manager, envía mediante una solicitud `POST` los datos de un cliente (correo electrónico y SKU de un producto eliminado) a un servicio web externo en `https://christusmuguerza.com.mx/kunmap/web_service.php`, indicando que la acción realizada es de tipo `remove` (eliminación del carrito).

Cuando el servidor responde exitosamente, el script muestra en la consola del navegador el contenido devuelto por el servicio.

Trigger Custom Event



6) API Receptora

Este script PHP actúa como **API de recepción** para datos enviados desde GTM (cuando un usuario agrega un producto al carrito). Hace validaciones, transforma y limpia los datos, genera un GUID único, construye el payload con IDs de campos que Eloqua espera, y finalmente envía esos datos al endpoint de un formulario en Eloqua.

Código (versión con comentarios)

```php
<?php
// 1. Configurar cabeceras CORS
header("Access-Control-Allow-Origin: *"); // Permite peticiones desde cualquier dominio
header("Access-Control-Allow-Methods: POST"); // Solo permite POST
header("Access-Control-Allow-Headers: Content-Type"); // Permite header Content-Type en la
petición

// 2. Log de datos crudos recibidos (para debugging)
file_put_contents('log.txt', file_get_contents('php://input') . "\n", FILE_APPEND);

// 3. Preflight check para peticiones OPTIONS (CORS)
if ($_SERVER['REQUEST_METHOD'] === 'OPTIONS') {
    http_response_code(204); // Sin contenido
    exit;
}

// 4. Validar que solo se aceptan peticiones POST
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    http_response_code(405); // Método no permitido
    echo json_encode(['error' => 'Only POST allowed']);
    exit;
}

// 5. Leer y decodificar JSON
$input = file_get_contents('php://input'); // Datos crudos del cuerpo
$data = json_decode($input, true); // Convertir JSON a array asociativo

// 6. Validación básica: debe existir 'email'
if (!$data || !isset($data['email'])) {
    http_response_code(400);
    echo json_encode(['error' => 'Invalid or missing data']);
    exit;
}

// 7. Generar un GUID único (8 caracteres hexadecimales)
$guid = 'id_' . substr(md5(uniqid("", true)), 0, 8);

// 8. Transformar timestamp a formato HH:MM:SS
if (!empty($data['timestamp'])) {
    try {
        $datetime = new DateTime($data['timestamp']);
        $datetime->setTimezone(new DateTimeZone('America/Mexico_City'));
        $data['timestamp'] = $datetime->format('H:i:s');
    } catch (Exception $e) {
        $data['timestamp'] = ''; // Fallback en caso de error
    }
}

// 9. Extraer el slug de la URL (última parte después de '/')
if (!empty($data['url'])) {
    $parsedUrl = parse_url($data['url'], PHP_URL_PATH);
    $data['url'] = basename($parsedUrl);
}

// 10. Extraer solo la parte de la imagen después de '/uploads/'
if (!empty($data['image_url'])) {
    $uploadsPos = strpos($data['image_url'], '/uploads/');
    if ($uploadsPos !== false) {
        $data['image_url'] = substr($data['image_url'], $uploadsPos + strlen('/uploads/'));
    }
```

```
}

// 11. Construir payload para Eloqua (usando IDs de campos del formulario)
$eloquaPayload = [
    "fieldValues" => [
        [ "id" => "6658", "value" => $data['email'] ],
        [ "id" => "6671", "value" => $data['service_name'] ?? '' ],
        [ "id" => "6672", "value" => $data['sku'] ?? '' ],
        [ "id" => "6673", "value" => $data['category'] ?? '' ],
        [ "id" => "6674", "value" => $data['quantity'] ?? '' ],
        [ "id" => "6675", "value" => $data['price'] ?? '' ],
        [ "id" => "6678", "value" => $data['image_url'] ?? '' ],
        [ "id" => "6873", "value" => $data['timestamp'] ?? '' ],
        [ "id" => "6872", "value" => $data['medical_unit'] ?? '' ],
        [ "id" => "6874", "value" => $data['url'] ?? '' ],
        [ "id" => "6875", "value" => $guid],
        [ "id" => "6876", "value" => $data['order'] ?? '' ],
        [ "id" => "6911", "value" => $data['pricem'] ?? '' ],
        [ "id" => "6912", "value" => $data['cita'] ?? '' ],
        [ "id" => "6936", "value" => $data['discount'] ?? '' ],
        [ "id" => "6984", "value" => $data['utmSource'] ?? '' ],
        [ "id" => "6986", "value" => $data['utmCampaign'] ?? '' ],
        [ "id" => "6985", "value" => $data['utmMedium'] ?? '' ],
    ]
];

// 12. Enviar a Eloqua vía API REST (endpoint de formulario)
$ch = curl_init("https://secure.p04.eloqua.com/api/REST/2.0/data/form/618");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, [
    "Content-Type: application/json",
    "Authorization: Basic Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM=" //
credenciales codificadas Base64
]);
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($eloquaPayload));

$response = curl_exec($ch);
$httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
curl_close($ch);

// 13. Responder al cliente según el resultado
if ($httpCode >= 200 && $httpCode < 300) {
    echo json_encode(["success" => true]);
} else {
    http_response_code($httpCode);
    echo json_encode(["error" => "Eloqua API error", "response" => $response]);
}
```

Explicación detallada

```
header("Access-Control-Allow-Origin: *");
header("Access-Control-Allow-Methods: POST");
header("Access-Control-Allow-Headers: Content-Type");
```

Permiten que el script sea llamado desde cualquier dominio `(*)` y solo aceptando `POST`. También permite el encabezado `Content-Type` para enviar JSON.

```
file_put_contents('log.txt', file_get_contents('php://input') . "\n", FILE_APPEND);
```

Guarda en `log.txt` todo el cuerpo crudo recibido (`php://input`) para depuración.

```
if ($_SERVER['REQUEST_METHOD'] === 'OPTIONS') {
    http_response_code(204); // Sin contenido
    exit;
}
```

Si el navegador envía una petición previa para verificar CORS, responde con `204 No Content` y termina. Esto evita errores en navegadores como Chrome.

```
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    http_response_code(405); // Método no permitido
    echo json_encode(['error' => 'Only POST allowed']);
    exit;
}
```

Si alguien intenta usar GET o PUT, se devuelve un `405 Method Not Allowed`.

```
$input = file_get_contents('php://input'); // Datos crudos del cuerpo
$data = json_decode($input, true); // Convertir JSON a array asociativo
if (!$data || !isset($data['email'])) {
    http_response_code(400);
    echo json_encode(['error' => 'Invalid or missing data']);
    exit;
}
```

Lee `php://input`, lo convierte en array asociativo y verifica que haya un campo `email`. Si falta o no es válido, retorna `400 Bad Request`.

```
$guid = 'id_' . substr(md5(uniqid("", true)), 0, 8);
```

Crea un identificador único de 8 caracteres hexadecimales con prefijo `id_`. Esto sirve para identificar de forma única cada envío a Eloqua.

```
if (!empty($data['timestamp'])) {
    try {
        $datetime = new DateTime($data['timestamp']);
        $datetime->setTimezone(new DateTimeZone('America/Mexico_City'));
        $data['timestamp'] = $datetime->format('H:i:s');
    } catch (Exception $e) {
        $data['timestamp'] = ''; // Fallback en caso de error
    }
}
```

Convierte la fecha/hora recibida a formato HH:MM:SS ajustada a la zona horaria America/Mexico_City. Si el formato es inválido, se asigna vacío.

```
if (!empty($data['url'])) {
    $parsedUrl = parse_url($data['url'], PHP_URL_PATH);
    $data['url'] = basename($parsedUrl);
}
```

Ejemplo: si url es `https://dominio.com/producto/resonancia-magnetica`, se queda solo con `resonancia-magnetica`.

```
if (!empty($data['image_url'])) {
    $uploadsPos = strpos($data['image_url'], '/uploads/');
    if ($uploadsPos !== false) {
        $data['image_url'] = substr($data['image_url'], $uploadsPos + strlen('/uploads/'));
    }
}
```

Toma únicamente la parte después de `/uploads/` para simplificar el almacenamiento en Eloqua.

```
$eloquaPayload = [
    "fieldValues" => [
        [ "id" => "6658", "value" => $data['email'] ],
        [ "id" => "6671", "value" => $data['service_name'] ?? '' ],
        [ "id" => "6672", "value" => $data['sku'] ?? '' ],
        [ "id" => "6673", "value" => $data['category'] ?? '' ],
        [ "id" => "6674", "value" => $data['quantity'] ?? '' ],
        [ "id" => "6675", "value" => $data['price'] ?? '' ],
        [ "id" => "6678", "value" => $data['image_url'] ?? '' ],
        [ "id" => "6873", "value" => $data['timestamp'] ?? '' ],
        [ "id" => "6872", "value" => $data['medical_unit'] ?? '' ],
        [ "id" => "6874", "value" => $data['url'] ?? '' ],
        [ "id" => "6875", "value" => $guid],
        [ "id" => "6876", "value" => $data['order'] ?? '' ],
        [ "id" => "6911", "value" => $data['pricem'] ?? '' ],
        [ "id" => "6912", "value" => $data['cita'] ?? '' ],
        [ "id" => "6936", "value" => $data['discount'] ?? '' ],
        [ "id" => "6984", "value" => $data['utmSource'] ?? '' ],
        [ "id" => "6986", "value" => $data['utmCampaign'] ?? '' ],
        [ "id" => "6985", "value" => $data['utmMedium'] ?? '' ],
    ]
];
```

- Cada campo se asigna a un **ID de campo** de Eloqua (form field ID).
- Si un valor no está presente, se asigna `''` (string vacío).
- Esto asegura que Eloqua reciba exactamente lo que espera.

```
$ch = curl_init("https://secure.p04.eloqua.com/api/REST/2.0/data/form/618");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, [
    "Content-Type: application/json",
```

```
    "Authorization: Basic Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM=" //
credenciales codificadas Base64
]);
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($eloquaPayload));

$response = curl_exec($ch);
$httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
curl_close($ch);
```

- `curl_init()` abre la conexión al endpoint del formulario en Eloqua (form/618).
- Se configuran headers para JSON y autenticación **Basic Auth** (usuario y contraseña codificados en Base64).
- `CURLOPT_POSTFIELDS` envía el cuerpo JSON.
- `CURLOPT_RETURNTRANSFER` hace que la respuesta se capture en `$response` y no se imprima directamente.

```
if ($httpCode >= 200 && $httpCode < 300) {
    echo json_encode(["success" => true]);
} else {
    http_response_code($httpCode);
    echo json_encode(["error" => "Eloqua API error", "response" => $response]);
}
```

- Si Eloqua responde con código 2xx → `{ "success": true }`.
- Si responde con otro código, se devuelve el mismo HTTP code y un JSON con el error y la respuesta original.

API de procesamiento

Esta API en PHP recibe solicitudes desde un sitio web (o Google Tag Manager) para registrar y administrar eventos de carrito de compras en **OIC** y **Eloqua**.

En resumen:

- **Entrada:** Recibe datos vía `POST` con un parámetro `type` (`cart` o `remove`) y otros campos como email, SKU, nombre del servicio, precio, etc.
- **Si** `type = cart`**:**
    1. Valida el email y procesa datos del producto (y opcionalmente productos relacionados).
    2. Genera un identificador único (`guid`).
    3. Obtiene un `access_token` desde **Oracle Identity Cloud (OIC)**.
    4. Envía los datos del carrito a un endpoint de **OIC** (`AddCart`).
    5. Si hay productos relacionados, los envía también a **Eloqua** usando la API de formularios.
- **Si** `type = remove`**:**
    1. Busca en **Eloqua** un registro del carrito en un *Custom Object* usando el email y SKU.
    2. Si lo encuentra, lo elimina mediante la API REST de Eloqua.

- Recoge datos de eventos de carrito.
- Los sincroniza con Oracle (OIC) y Eloqua.

- También permite eliminar registros del carrito en Eloqua.

Código versión con comentarios

```php
<?php
header("Access-Control-Allow-Origin: *");
header('Access-Control-Allow-Methods: POST, GET, OPTIONS');
header('Access-Control-Max-Age: 1000');
header('Access-Control-Allow-Headers: Content-Type, *');

$type = $_POST["type"];
$email = $_POST["email"];

// Validar si $email está vacío o no contiene un formato de correo electrónico válido
if (empty($email) || !filter_var($email, FILTER_VALIDATE_EMAIL)) {
    // Detener la ejecución del script y devolver un mensaje de error
    exit();

}

if ($type == "cart") {
    $service_name = $_POST["service_name"];
    $sku = $_POST["sku"];
    $category = $_POST["category"];
    $quantity = $_POST["quantity"];
    $price = $_POST["price"];
    $image_url = $_POST["image_url"];
    $timestamp = $_POST["timestamp"];
    $medical_unit = $_POST["medical_unit"];
    $url = $_POST["url"];
    $order = $_POST["order"];
    $pricem = $_POST["pricem"];
    $cita = $_POST["cita"];
    $discount = $_POST["discount"];
    $p_related = $_POST["p_related"];
    $utmSource = $_POST["utmSource"];
    $utmMedium = $_POST["utmMedium"];
    $utmCampaign = $_POST["utmCampaign"];

    //Validadión de campos
    $discount_v = $pricem !== "" ? 'true' : '';
    $cita_v = ($cita !== "" && $cita !== "undefined") ? 'true' : '';

    if ($discount == "Membresía") {
        $membresia_v = 'true';
        $oferta_v = '';
    } else if ($discount == "¡Oferta!") {
        $oferta_v = 'true';
        $membresia_v = '';
    } else {
        $membresia_v = '';
        $oferta_v = '';
        $discount = '';
    }

    // Generar un GUID
```

```
    $guid = 'id_' . substr(md5(uniqid("", true)), 0, 8); // Ajusta la longitud según tus
necesidades


    //Autenticación OIC
    $curl = curl_init();

    curl_setopt_array($curl, array(
        CURLOPT_URL => 'https://idcs-
8332050b9ca94ab48f84d174e8db9675.identity.oraclecloud.com/oauth2/v1/token',
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => '',
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 0,
        CURLOPT_FOLLOWLOCATION => true,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => 'POST',
        CURLOPT_POSTFIELDS =>
'grant_type=client_credentials&client_id=8c4e30a91aed4b68a4ed0994d4a18f8c&client_secret=d16a9
8e1-5ccf-4c13-9fb8-2495157cbf81&scope=christusmuguerza.com.mx%2FEcommerce',
        CURLOPT_HTTPHEADER => array(
            'Content-Type: application/x-www-form-urlencoded'
        ),
    ));

    $response = curl_exec($curl);
    curl_close($curl);
    $access_token = json_decode($response)->access_token;




    // Enviar datos a OIC
    $curl = curl_init();
    curl_setopt_array($curl, array(
        CURLOPT_URL => 'https://servicios-oic.christus.mx/CloudUtilities/Ecommerce/AddCart',
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => '',
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 0,
        CURLOPT_FOLLOWLOCATION => true,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => 'POST',
        CURLOPT_POSTFIELDS => '{
          "Header": {
            "TrackingID": "ECOMCART",
            "Source": "ECOMMERCE"
          },
          "File": {
            "Auth" : "AUTH",
            "Date" : "' . $timestamp . '",
            "type": "FormData",
            "fieldValues":  [
                {
                    "type": "FieldValue",
                    "id": "6658",
                    "value": "' . $email . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6671",
                    "value": "' . $service_name . '"
                },
```

```
                {
                    "type": "FieldValue",
                    "id": "6672",
                    "value": "' . $sku . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6673",
                    "value": "' . $category . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6674",
                    "value": "' . $quantity . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6675",
                    "value": "' . $price . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6911",
                    "value": "' . $pricem . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6678",
                    "value": "' . $image_url . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6873",
                    "value": "' . $timestamp . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6872",
                    "value": "' . $medical_unit . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6874",
                    "value": "' . $url . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6875",
                    "value": "' . $guid . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6876",
                    "value": "' . $order . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6912",
                    "value": "' . $cita . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6936",
```

```
                    "value": "' . $discount . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6939",
                    "value": "' . $discount_v . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6933",
                    "value": "' . $cita_v . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6938",
                    "value": "' . $membresia_v . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6937",
                    "value": "' . $oferta_v . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6984",
                    "value": "' . $utmSource . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6985",
                    "value": "' . $utmMedium . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6986",
                    "value": "' . $utmCampaign . '"
                }
            ]
          }
        }',
        CURLOPT_HTTPHEADER => array(
            'Content-Type: application/json',
            'Authorization: Bearer ' . $access_token
        ),
    ));

    $response = curl_exec($curl);
    curl_close($curl);

    // $logFilePath = 'respuesta_log.txt';
    // $logFile = fopen($logFilePath, 'a+');
    // fwrite($logFile, date('Y-m-d H:i:s') . " - Respuesta: CART\n " . $response . "\n\n");
    // fclose($logFile);
    echo $response;


    if ($p_related != null) {
        $parsed_related = json_decode($p_related, true);

        foreach ($parsed_related as $product) {
            $name_pr = $product['name'];
            $price_pr = $product['price'];
            $image_url_pr = $product['image_url'];
```

```
            $url_pr = $product['product_url'];
            $order_pr = $product['order'];
            $sale_pr = $product['sale'];

            // Enviar datos a Eloqua
            $curl = curl_init();
            curl_setopt_array($curl, array(
                CURLOPT_URL => 'https://secure.p04.eloqua.com/API/REST/2.0/data/form/646',
                CURLOPT_RETURNTRANSFER => true,
                CURLOPT_ENCODING => '',
                CURLOPT_MAXREDIRS => 10,
                CURLOPT_TIMEOUT => 0,
                CURLOPT_FOLLOWLOCATION => true,
                CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
                CURLOPT_CUSTOMREQUEST => 'POST',
                CURLOPT_POSTFIELDS => '{
            "type": "FormData",
            "fieldValues": [
                {
                    "type": "FieldValue",
                    "id": "6955",
                    "value": "' . $email . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6956",
                    "value": "' . $name_pr . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6957",
                    "value": "' . $category . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6958",
                    "value": "' . $price_pr . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6960",
                    "value": "' . $image_url_pr . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6959",
                    "value": "' . $timestamp . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6962",
                    "value": "' . $medical_unit . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6964",
                    "value": "' . $url_pr . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6954",
                    "value": "' . $guid . $order_pr . '"
                },
```

```
                {
                    "type": "FieldValue",
                    "id": "6961",
                    "value": "' . $order_pr . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6967",
                    "value": "' . $sale_pr . '"
                }
            ]
        }',
                CURLOPT_HTTPHEADER => array(
                    'Content-Type: application/json',
                    'Authorization: Basic
Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM='
                ),
            ));

            $response = curl_exec($curl);
            curl_close($curl);

            // $logFilePath = 'respuesta_log.txt';
            // $logFile = fopen($logFilePath, 'a+');
            // fwrite($logFile, date('Y-m-d H:i:s') . " - Respuesta PR:\n " . $response .
"\n\n");
            // fclose($logFile);
            // echo $response;
        }
    }
} else if ($type == "remove") {

    $email = $_POST["email"];
    $sku = $_POST["sku"];

    // URL encode
    $encodedEmail = urlencode($email);
    $encodedSku = urlencode($sku);

    $url =
'https://secure.p04.eloqua.com/api/REST/2.0/data/customObject/271/instances?search=SKU1%3D%27
' . $encodedSku . '%27Email_Address1%3D%27' . $encodedEmail . '%27';

    $curl = curl_init();
    curl_setopt_array($curl, array(
        CURLOPT_URL => $url,
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => '',
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 0,
        CURLOPT_FOLLOWLOCATION => true,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => 'GET',
        CURLOPT_HTTPHEADER => array(
            'Authorization: Basic Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM='
        ),
    ));

    $response = curl_exec($curl);
    curl_close($curl);
    echo $response;

    // Log the GET request and response
```

```
    // $logFile = __DIR__ . '/log.txt';

    $elements = json_decode($response)->elements ?? [];
    foreach ($elements as $element) {
        $id = $element->id;
        $deleteUrl =
'https://secure.p04.eloqua.com/api/REST/2.0/data/customObject/271/instance/' . $id;
        $curl = curl_init();
        curl_setopt_array($curl, array(
            CURLOPT_URL => $deleteUrl,
            CURLOPT_RETURNTRANSFER => true,
            CURLOPT_ENCODING => '',
            CURLOPT_MAXREDIRS => 10,
            CURLOPT_TIMEOUT => 0,
            CURLOPT_FOLLOWLOCATION => true,
            CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
            CURLOPT_CUSTOMREQUEST => 'DELETE',
            CURLOPT_HTTPHEADER => array(
                'Authorization: Basic
Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM='
            ),
        ));
        $response = curl_exec($curl);
        curl_close($curl);
        echo $response;

        // $logEntry = "\n$encodedEmail\n DELETE Request: $deleteUrl\nResponse: $response\n";
        // file_put_contents($logFile, $logEntry, FILE_APPEND);
    }
}
```

Explicación detallada

```
header("Access-Control-Allow-Origin: *");
header('Access-Control-Allow-Methods: POST, GET, OPTIONS');
header('Access-Control-Max-Age: 1000');
header('Access-Control-Allow-Headers: Content-Type, *');
```

- Permite que cualquier origen (dominio) consuma la API.
- Declara que aceptas POST, GET y OPTIONS (útil para preflight en navegadores).
- Autoriza el header Content-Type y otros.

```
$type = $_POST["type"];
$email = $_POST["email"];

if (empty($email) || !filter_var($email, FILTER_VALIDATE_EMAIL)) {
    exit();

}
```

- Lee `type` (control de flujo) y `email` (clave en los envíos).
- Valida que `email` no venga vacío y tenga formato válido; si falla, termina el script sin responder nada (solo cierra).

```
if ($type == "cart") {
    $service_name = $_POST["service_name"];
    $sku = $_POST["sku"];
    $category = $_POST["category"];
    $quantity = $_POST["quantity"];
    $price = $_POST["price"];
    $image_url = $_POST["image_url"];
    $timestamp = $_POST["timestamp"];
    $medical_unit = $_POST["medical_unit"];
    $url = $_POST["url"];
    $order = $_POST["order"];
    $pricem = $_POST["pricem"];
    $cita = $_POST["cita"];
    $discount = $_POST["discount"];
    $p_related = $_POST["p_related"];
    $utmSource = $_POST["utmSource"];
    $utmMedium = $_POST["utmMedium"];
    $utmCampaign = $_POST["utmCampaign"];
```

Cuando type es "cart", se asume que estás registrando un **evento de agregar al carrito**.

- Recolección de campos de carrito: todos estos llegan del cliente (GTM, frontend, etc.).
- `p_related` es un **JSON** con productos relacionados (si viene).

```
$discount_v = $pricem !== "" ? 'true' : '';
    $cita_v = ($cita !== "" && $cita !== "undefined") ? 'true' : '';

    if ($discount == "Membresía") {
        $membresia_v = 'true';
        $oferta_v = '';
    } else if ($discount == "¡Oferta!") {
        $oferta_v = 'true';
        $membresia_v = '';
    } else {
        $membresia_v = '';
        $oferta_v = '';
        $discount = '';
    }
```

- `discount_v`: queda `'true'` si existe `pricem` (precio de membresía).
- `cita_v`: queda `'true'` si cita no está vacío ni es la cadena `"undefined"`.
- Normaliza `discount` en dos flags: `membresia_v` u `oferta_v`. Si no coincide, limpia `discount`.

```
$guid = 'id_' . substr(md5(uniqid("", true)), 0, 8);
```

- Crea un identificador pseudoúnico corto para asociar el evento/registro.

```
$curl = curl_init();

    curl_setopt_array($curl, array(
```

```
        CURLOPT_URL => 'https://idcs-
8332050b9ca94ab48f84d174e8db9675.identity.oraclecloud.com/oauth2/v1/token',
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => '',
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 0,
        CURLOPT_FOLLOWLOCATION => true,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => 'POST',
        CURLOPT_POSTFIELDS =>
'grant_type=client_credentials&client_id=8c4e30a91aed4b68a4ed0994d4a18f8c&client_secret=d16a98e1-
5ccf-4c13-9fb8-2495157cbf81&scope=christusmuguerza.com.mx%2FEcommerce',
        CURLOPT_HTTPHEADER => array(
            'Content-Type: application/x-www-form-urlencoded'
        ),
    ));

    $response = curl_exec($curl);
    curl_close($curl);
    $access_token = json_decode($response)->access_token;

$curl = curl_init();
    curl_setopt_array($curl, array(
        CURLOPT_URL => 'https://servicios-oic.christus.mx/CloudUtilities/Ecommerce/AddCart',
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => '',
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 0,
        CURLOPT_FOLLOWLOCATION => true,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => 'POST',
        CURLOPT_POSTFIELDS => '{
          "Header": {
            "TrackingID": "ECOMCART",
            "Source": "ECOMMERCE"
          },
          "File": {
            "Auth" : "AUTH",
            "Date" : "' . $timestamp . '",
            "type": "FormData",
            "fieldValues":  [
                {
                    "type": "FieldValue",
                    "id": "6658",
                    "value": "' . $email . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6671",
                    "value": "' . $service_name . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6672",
                    "value": "' . $sku . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6673",
                    "value": "' . $category . '"
                },
                {
                    "type": "FieldValue",
```

```
                    "id": "6674",
                    "value": "' . $quantity . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6675",
                    "value": "' . $price . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6911",
                    "value": "' . $pricem . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6678",
                    "value": "' . $image_url . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6873",
                    "value": "' . $timestamp . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6872",
                    "value": "' . $medical_unit . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6874",
                    "value": "' . $url . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6875",
                    "value": "' . $guid . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6876",
                    "value": "' . $order . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6912",
                    "value": "' . $cita . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6936",
                    "value": "' . $discount . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6939",
                    "value": "' . $discount_v . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6933",
                    "value": "' . $cita_v . '"
                },
```

```
                {
                    "type": "FieldValue",
                    "id": "6938",
                    "value": "' . $membresia_v . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6937",
                    "value": "' . $oferta_v . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6984",
                    "value": "' . $utmSource . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6985",
                    "value": "' . $utmMedium . '"
                },
                {
                    "type": "FieldValue",
                    "id": "6986",
                    "value": "' . $utmCampaign . '"
                }
            ]
          }
        }',
        CURLOPT_HTTPHEADER => array(
            'Content-Type: application/json',
            'Authorization: Bearer ' . $access_token
        ),
    ));

    $response = curl_exec($curl);
    curl_close($curl);

    echo $response;
```

- Se construye un JSON con un **Header** y un cuerpo **File** que encapsula un array `fieldValues` con **IDs de Eloqua** (o mapeos internos) y sus valores.
- Se autoriza con `Bearer {access_token}` obtenido antes.
- Se devuelve (`echo`) la respuesta de OIC al cliente.

```
if ($p_related != null) {
        $parsed_related = json_decode($p_related, true);

        foreach ($parsed_related as $product) {
            $name_pr = $product['name'];
            $price_pr = $product['price'];
            $image_url_pr = $product['image_url'];
            $url_pr = $product['product_url'];
            $order_pr = $product['order'];
            $sale_pr = $product['sale'];

            // Enviar datos a Eloqua
            $curl = curl_init();
            curl_setopt_array($curl, array(
                CURLOPT_URL => 'https://secure.p04.eloqua.com/API/REST/2.0/data/form/646',
                CURLOPT_RETURNTRANSFER => true,
```

```
            CURLOPT_ENCODING => '',
            CURLOPT_MAXREDIRS => 10,
            CURLOPT_TIMEOUT => 0,
            CURLOPT_FOLLOWLOCATION => true,
            CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
            CURLOPT_CUSTOMREQUEST => 'POST',
            CURLOPT_POSTFIELDS => '{
        "type": "FormData",
        "fieldValues": [
            {
                "type": "FieldValue",
                "id": "6955",
                "value": "' . $email . '"
            },
            {
                "type": "FieldValue",
                "id": "6956",
                "value": "' . $name_pr . '"
            },
            {
                "type": "FieldValue",
                "id": "6957",
                "value": "' . $category . '"
            },
            {
                "type": "FieldValue",
                "id": "6958",
                "value": "' . $price_pr . '"
            },
            {
                "type": "FieldValue",
                "id": "6960",
                "value": "' . $image_url_pr . '"
            },
            {
                "type": "FieldValue",
                "id": "6959",
                "value": "' . $timestamp . '"
            },
            {
                "type": "FieldValue",
                "id": "6962",
                "value": "' . $medical_unit . '"
            },
            {
                "type": "FieldValue",
                "id": "6964",
                "value": "' . $url_pr . '"
            },
            {
                "type": "FieldValue",
                "id": "6954",
                "value": "' . $guid . $order_pr . '"
            },
            {
                "type": "FieldValue",
                "id": "6961",
                "value": "' . $order_pr . '"
            },
            {
                "type": "FieldValue",
                "id": "6967",
                "value": "' . $sale_pr . '"
```

```
                }
            ]
        }',
                CURLOPT_HTTPHEADER => array(
                    'Content-Type: application/json',
                    'Authorization: Basic
Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM='
                ),
            ));

            $response = curl_exec($curl);
            curl_close($curl);

            // $logFilePath = 'respuesta_log.txt';
            // $logFile = fopen($logFilePath, 'a+');
            // fwrite($logFile, date('Y-m-d H:i:s') . " - Respuesta PR:\n " . $response .
"\n\n");
            // fclose($logFile);
            // echo $response;
        }
    }
```

- Si `p_related` trae una lista, por **cada producto** se hace un `POST` a la API de Forms de Eloqua (`/API/REST/2.0/data/form/646`).
- Se incluye `email`, `name`, `category` (heredada), `price`, `image_url`, `timestamp`, `medical_unit`, `product_url`, un GUID derivado (`$guid.$order_pr`), `order` y `sale`.
- Autenticación **Basic** con credencial codificada en Base64 (usuario:contraseña de Eloqua).

```
} else if ($type == "remove") {

    $email = $_POST["email"];
    $sku = $_POST["sku"];

    // URL encode
    $encodedEmail = urlencode($email);
    $encodedSku = urlencode($sku);

    $url =
'https://secure.p04.eloqua.com/api/REST/2.0/data/customObject/271/instances?search=SKU1%3D%27' .
$encodedSku . '%27Email_Address1%3D%27' . $encodedEmail . '%27';

    $curl = curl_init();
    curl_setopt_array($curl, array(
        CURLOPT_URL => $url,
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => '',
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 0,
        CURLOPT_FOLLOWLOCATION => true,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => 'GET',
        CURLOPT_HTTPHEADER => array(
            'Authorization: Basic Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM='
        ),
    ));

    $response = curl_exec($curl);
    curl_close($curl);
```

```
    echo $response;

    // Log the GET request and response
    // $logFile = __DIR__ . '/log.txt';

    $elements = json_decode($response)->elements ?? [];
    foreach ($elements as $element) {
        $id = $element->id;
        $deleteUrl = 'https://secure.p04.eloqua.com/api/REST/2.0/data/customObject/271/instance/'
. $id;
        $curl = curl_init();
        curl_setopt_array($curl, array(
            CURLOPT_URL => $deleteUrl,
            CURLOPT_RETURNTRANSFER => true,
            CURLOPT_ENCODING => '',
            CURLOPT_MAXREDIRS => 10,
            CURLOPT_TIMEOUT => 0,
            CURLOPT_FOLLOWLOCATION => true,
            CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
            CURLOPT_CUSTOMREQUEST => 'DELETE',
            CURLOPT_HTTPHEADER => array(
                'Authorization: Basic Q2hyaXN0dXNcS3VubWFwLkFnZW5jaWE6S3VuTTRwS3VuTTRwQDIwMjM='
            ),
        ));
        $response = curl_exec($curl);
        curl_close($curl);
        echo $response;

        // $logEntry = "\n$encodedEmail\n DELETE Request: $deleteUrl\nResponse: $response\n";
        // file_put_contents($logFile, $logEntry, FILE_APPEND);
    }
}
```

Cuando `type` es `"remove"`, se asume que quieres **eliminar** un registro en Eloqua (Custom Object 271) asociado a un `email` y un `sku`.

- Prepara una llamada GET al **Custom Object 271** buscando instancias que hagan match con `SKU1 = 'sku'` y `Email_Address1 = 'email'`.
- Con las coincidencias encontradas, se **itera** y hace un `DELETE` por id de cada instancia.
- Autenticación: **Basic** con las mismas credenciales de Eloqua.
- `echo` de las respuestas (tanto del GET como del DELETE).