iCredit Payment Gateway Implementation

General :

iCredit is a credit card payment page gateway, the page can be used as a payment methods when checking out from the cart in the ecommerce application.

The payment page can show in 3 ways :

1. Redirect – the client will be redirected out of the ecommerce site to the payment page, and after payment will be redirected back to the ecommerce site. (Like Paypal)
2. iFrame – the payment page will show in an iframe in the ecommerce site.
3. Popup – the payment page will show in an iframe in a modal over the ecommerce site.

Technology :

REST API Service using JSON.

Test Evnoirment : https://testicredit.rivhit.co.il/API/PaymentPageRequest.svc/GetUrl
Prod Evnoirment : https://icredit.rivhit.co.il/API/PaymentPageRequest.svc/GetUrl

Test GroupPrivateToken : XXXXXX (you will receive the token later)

Flow :

After the plugin will be installed and configured in the ecommerce application, the work flow should look like this :

1. The customer enters the site and adds items to his cart.
2. When finished, the customer clicks on the "checkout" button to confirm the cart items.
3. The customer may enter a discount coupon if he has one.
4. The customer select the "iCredit credit card" payment option and press "Pay" (Order status is marked "Processing") (* optional – if the site has more then one payment method)
5. The ecommerce sends a request through the API to get a payment page URL.
6. The ecommerce redirect (or show in an iframe) the customer to the payment page
7. The customer enter his credit card information in the payment page and charges the card.
8. After a successful charge the customer is redirected back to the ecommerce to a Thank You Page
9. At the same time our server send an IPN (Instant Payment Notification) message to the ecommerce site with all of the transaction details. (Order status is marked "Completed")
    a. If an error occurred in the IPN process, the Order status is marked "On Hold"

Design :

Backoffice definition form :

Form Fields :

Field Name	Control Type	API Parameter	Values / Logica
Enable / Disable	CheckBox	-	Enable the payment method in the store
Payment Method Description	Input	-	The paying method description in the store
Test Mode	CheckBox	-	If checked then work with the **Test URL** otherwise work with the **Prod URL**
Test Group Token	Input	GroupPrivateToken	If Test Mode is checked send this value
Prod Group Token	Input	GroupPrivateToken	If Test Mode is NOT checked send this value
Max Payments	Int	MaxPayments	
Credit from Payment	Int	CreditFromPayment	
Window Type	DropDown List	-	Redirect iFrame PopUp
iFrame/Popup Height	Input	-	the height in pixels
iFrame/Popup Width	Input	-	the width in pixels
Hide Items List	CheckBox	HideItemList	Checked = true / Uncheck = false
Create Token	CheckBox	CreateToken	Checked = true / Uncheck = false
Invoice Language	DropDown List	DocumentLanguage	Always Hebrew Always English By Billing Address By Shipping Address By Shipping or Billing Address (if the country code is IL then Hebrew otherwise English)
Exempt VAT	DropDown List	ExemptVAT	Always Charge VAT Always Exempt By Billing Address By Shipping Address By Shipping or Billing Address (if the country code is IL then Charge VAT otherwise Exempt)
Redirect URL	Input	RedirectURL	Can be internal and not a field
IPN URL	Input	IPNURL	Can be internal and not a field

Payment Button Details

When the client press the "Pay" button the application need to send the following data to the API.

Customer Details : (if any field does not exsixts please notify me)

Field	API Parameter	Data type	Comments
Company Name / Last Name	CustomerLastName	String	
First Name	CustomerFirstName	String	
Email	EmailAddress	String	
Address	Address	String	
City	City	String	
Zipcode	Zipcode	Int	Only if country code is IL
P.O. Box	POB	Int	
Phone Number	PhoneNumber	String	
Phone Number 2	PhoneNumber2	String	
Fax Number	FaxNumber	String	
Id Number	IdNumber	Int	Only if country code is IL
Vat Number	VatNumber	Int	Only if country code is IL
Comments	Comments	String	

Items Details : (For each item in the cart)

Field	API Parameter	Data type	Comments
0	Id	Int	Always = 0
Catalog Number / SKU	CatalogNumber	String	
Unit Price	UnitPrice	Decimal	
Quantity	Quantity	Decimal	
Item Description	Description	String	

Sale Details :

Field	API Parameter	Data type	Comments
Total Discount Amount	Discount	Decimal	
Custom Field	Custom1-9	String	Can be used to pass the order id in the ecommerce and receive it back from the IPN

Pay Button Algorithm :

Here is a complete flow that activates when the client press the "Pay" button in the cart checkout page :

```
If  "Test Mode" = true
        URL = "https://testicredit.rivhit.co.il/API/PaymentPageRequest.svc/GetUrl"
        GroupPrivateToken = " Test Group Token "
Else
        URL = "https://icredit.rivhit.co.il/API/PaymentPageRequest.svc/GetUrl"
        GroupPrivateToken = " Prod Group Token "
End if

MaxPayments = " Max Payments "
CreditFromPayment = "Credit from Payment "
HideItemList = ("Hide Items List " == checked)
CreateToken = ("Create Token" == checked)

Case " Invoice Language "
        When " Always Hebrew"
                DocumentLanguage = "he"
        When " Always English "
                DocumentLanguage = "en"
        When " By Billing Address "
                If (Billing Address Country Code = IL)
                        DocumentLanguage = "he"
                Else
                        DocumentLanguage = "en"
        When " By Shipping Address"
                If (Shipping Address Country Code = IL)
                        DocumentLanguage = "he"
                Else
                        DocumentLanguage = "en"
        When " By Shipping or Billing Address"
                If (Billing Address Country Code = IL or Shipping Address Country Code = IL)
                        DocumentLanguage = "he"
                Else
                        DocumentLanguage = "en"


Case " Exempt VAT "
        When " Always Charge VAT "
                ExemptVAT = false
        When " Always Exempt "
                ExemptVAT = true
        When " By Billing Address "
                If (Billing Address Country Code = IL)
                        ExemptVAT = false
                Else
                        ExemptVAT = true
```

When " By Shipping Address"
If (Shipping Address Country Code = IL)
ExemptVAT = false
Else
ExemptVAT = true
When " By Shipping or Billing Address"
If (Billing Address Country Code = IL or Shipping Address Country Code = IL)
ExemptVAT = false
Else
ExemptVAT = true

RedirectURL = " Redirect URL "
IPNURL = "IPN URL"

CustomerLastName = " Company Name / Last Name "
CustomerFirstName = " First Name "
EmailAddress = " Email "
Address = " Address "
City = " City "
POB = " P.O. Box "
PhoneNumber = " Phone Number "
PhoneNumber2 = " Phone Number 2"
FaxNumber = " Fax Number "
If (Billing Address Country Code = IL)
Zipcode = " Zipcode "
IdNumber = " Id Number "
VatNumber = " Vat Number "
Comments = " Comments "

For each item in cart
Id = 0
CatalogNumber = "SKU "
UnitPrice = " Unit Price "
Quantity = " Quantity "
Description = " Item Description "

Discount = " Total Discount Amount "

Custom1 = "ecommerce order id" // you will get this value in the IPN to update the order
// status

POST data to URL to receive the "Payment Page URL"

Show the returned URL by " Window Type " (Redirect,iFrame,PopUp)

Mark the Order status = "Processing"

IPN (Instant Payment Notification)

After the customer has charged his credit card, we need to update the ecommerce application that the sale is completed.

This is done with an IPN Listener. The IPN Listener URL is posted to the GetURL payment page request in the prior step (the parameter is called "IPNURL").

Our server will send the following data to that URL using **POST** method :

Parameter	Notes
SaleId	
GroupPrivateToken	
NumberOfItmes	
ItemIdN	1 ≤ N ≤ NumberOfItmes
ItemCatalogNumberN	1 ≤ N ≤ NumberOfItmes
ItemQuantityN	1 ≤ N ≤ NumberOfItmes
ItemUnitPriceN	1 ≤ N ≤ NumberOfItmes
ItemDescriptionN	1 ≤ N ≤ NumberOfItmes
Reference	
Order	
CustomerLastName	
CustomerFirstName	
Address	
POB	
City	
Zipcode	
PhoneNumber	
PhoneNumber2	
FaxNumber	
IdNumber	
VatNumber	
Custom1	
Custom2	
Custom3	
Custom4	
Custom5	
Custom6	
Custom7	
Custom8	
Custom9	
CustomerId	
DocumentURL	
DocumentNum	
DocumentType	
TransactionAmount	
TransactionAuthNum	
TransactionCardName	
TransactionCardNum	
TransactionCreditTerms	
TransactionCreditTermsDescription	
TransactionFirstAmount	
TransactionNonFirstAmount	

TransactionNumOfPayment	
TransactionForeignSign	
TransactionSolekSapak	
TransactionStatus	
TransactionTerminalName	
TransactionTerminalNum	
TransactionTransAmount	
TransactionDateTime	
TransactionType	
TransactionTypeDescription	
TransactionCurrency	
TransactionCurrencyDescription	
TransactionToken	

The first thing the IPN Listener needs to do is to verify that the sale was charged in the payment page. This is done be sending a Verify API request.

You will need to implement an IPN Listener.
I have an implementation of the IPN in c#, you can base on that algorithm – see appendix B.

Keep in mind that the Order Id in the ecommerce application is in the Custome1 parameter (you send him when you request the payment page and get it back in the IPN).

After the IPN has verified and saved the payment details the Order status is marked "Completed".

If there was an error in the IPN the Order status is marked "On Hold".

"Thank You Page" Transition Page – for iframe / Popup only

As written above – after the customer successfully charch his credit card he is redirected to a Thank you page, which is passed to the GetURL as a parameter in "RedirectURL".

When using iframe or popup (modal) the redirecting will cause only the iFrame to redirect and not the parent window to redirect.
Because the redirecting page (the payment page) is not in the same domain, redirecting the parent window is not possible.

Therefore we need to redirect to a "Transition page" which is in the same domain, and it could then redirect the parent window.

Here is the javascript the "Transition page" should hold :

```
<script type="text/javascript">
    jQuery(document).ready(function () {

        window.top.location.href = "<%=ThankYouPageURL%>";

    });
</script>
```

Appendix A : Get URL full JSON request.


```
{
  "GroupPrivateToken":"1627aea5-8e0a-4371-9022-9b504344e724",
  "Items":[{
    "Id":0,
    "CatalogNumber":"String content",
    "UnitPrice":1234567.89,
    "Quantity":1234567.89,
    "Description":"String content"
  }],
  "RedirectURL":"String content",
  "IPNURL":"String content",
  "ExemptVAT":true,
  "MaxPayments":9999,
  "CreditFromPayment":9999,
  "EmailAddress":"String content",
  "CustomerLastName":"String content",
  "CustomerFirstName":"String content",
  "Address":"String content",
  "POB":12345,
  "City":"String content",
  "Zipcode":1234567,
  "PhoneNumber":"String content",
  "PhoneNumber2":"String content",
  "FaxNumber":"String content",
  "IdNumber":123456789,
  "VatNumber":123456789,
  "Comments":"String content",
  "HideItemList":true,
  "DocumentLanguage":"String content",
  "CreateToken":true,
  "Discount":1234567.89,
  "Custom1":"String content",
  "Custom2":"String content",
  "Custom3":"String content",
  "Custom4":"String content",
  "Custom5":"String content",
  "Custom6":"String content",
  "Custom7":"String content",
  "Custom8":"String content",
  "Custom9":"String content",
  "Reference":123456789,
  "Order":"String content",
  "EmailBcc":"String content",
  "CustomerId":123456789,
  "AgentId":123456789,
  "ProjectId":1234567
}
```

Appendix B : IPN Listener implementation in c#

```
<%@ Page Language="C#" %>

<%@ Import Namespace="System" %>
<%@ Import Namespace="System.Collections.Generic" %>
<%@ Import Namespace="System.Linq" %>
<%@ Import Namespace="System.Web" %>
<%@ Import Namespace="System.Web.UI" %>
<%@ Import Namespace="System.Web.UI.WebControls" %>
<%@ Import Namespace="System.IO" %>
<%@ Import Namespace="System.Net" %>
<%@ Import Namespace="System.Runtime.Serialization.Json" %>
<%@ Import Namespace="System.Text" %>
<%@ Import Namespace="System.Runtime.Serialization" %>
<script language="C#" option="Explicit" runat="server">
    [DataContract]
    public class VerifyRequestDTO
    {
        [DataMember]
        public Guid GroupPrivateToken { get; set; }
        [DataMember]
        public Guid SaleId { get; set; }
        [DataMember]
        public decimal TotalAmount { get; set; }
    }

    [DataContract]
    public class VerifyResponseDTO
    {
        [DataMember]
        public string Status { get; set; }
    }

    protected void Page_Load(object sender, EventArgs e)
    {
        if (!string.IsNullOrEmpty(Request.Params["SaleId"]))
        {
            string status = VerifyResponse(Guid.Parse(Request.Params["SaleId"]),
Guid.Parse(Request.Params["GroupPrivateToken"]),
decimal.Parse(Request.Params["TransactionAmount"]));

            if (status == "VERIFIED")
            {
                // TODO: Check that GroupId belongs to me
                // TODO: Check that transaction has not been previously processed
                // TODO: Check that amounts are correct

                // Example: writing all parameters to file on C:\temp\{SALEID}.txt
                using (FileStream fs = File.Create(@"C:\temp\" + Request.Params["SaleId"] +
".txt"))
                {
                    using (StreamWriter sw = new StreamWriter(fs))
                    {
                        sw.WriteLine("{0}", status);
                        foreach (string key in Request.Params.AllKeys)
                        {
                            sw.WriteLine("{0}: {1}", key, Request.Params[key]);
                        }
                    }
                }
            }
        }
```

```
    private string VerifyResponse(Guid saleId, Guid privateGroupToken, decimal amount)
    {
        VerifyRequestDTO parameters = new VerifyRequestDTO()
        {
            GroupPrivateToken = privateGroupToken,
            SaleId = saleId,
            TotalAmount = amount,
        };
        try
        {
            string VERIFY_TEST =
@"https://testicredit.rivhit.co.il/API/PaymentPageRequest.svc/Verify";
            string VERIFY_PROD =
@"https://icredit.rivhit.co.il/API/PaymentPageRequest.svc/Verify";
            HttpWebRequest req = WebRequest.Create(new Uri(VERIFY_TEST)) as HttpWebRequest;
            req.Method = "POST";
            req.ContentType = "application/json";
            req.Accept = "application/json";
            MemoryStream stream1 = new MemoryStream();
            DataContractJsonSerializer ser = new
DataContractJsonSerializer(typeof(VerifyRequestDTO));
            ser.WriteObject(stream1, parameters);
            stream1.Position = 0;
            StreamReader sr = new StreamReader(stream1);

            byte[] requestData = UTF8Encoding.UTF8.GetBytes(sr.ReadToEnd());
            req.ContentLength = requestData.Length;

            // Send the request:
            using (Stream post = req.GetRequestStream())
            {
                post.Write(requestData, 0, requestData.Length);
            }

            // Pick up and deserialize the response:
            DataContractJsonSerializer responseSerializer = new
DataContractJsonSerializer(typeof(VerifyResponseDTO));
            VerifyResponseDTO response;
            try
            {
                using (HttpWebResponse resp = req.GetResponse() as HttpWebResponse)
                {
                    if (resp.StatusCode == HttpStatusCode.OK)
                    {
                        response =
(VerifyResponseDTO)responseSerializer.ReadObject(resp.GetResponseStream());
                        return response.Status;
                    }
                    else
                    {
                        // The data is empty
                        response = null;
                        return "EXCEPTION";
                    }
                }
            }
            catch (WebException wex)
            {
                using (HttpWebResponse errorResponse = (HttpWebResponse)wex.Response)
                {
                    response =
(VerifyResponseDTO)responseSerializer.ReadObject(errorResponse.GetResponseStream());
                }
                return "EXCEPTION";
            }
        }
        catch (Exception)
        {
            return "EXCEPTION";
        }
    }
</script>
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head id="Head1" runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        Load this first to check the syntax of your page
    </div>
    </form>
</body>
</html>
```