WebLINK

3.0 Developer API Guide

BridgePay Gateway Edition

WebLINK Developer API Guide

BridgePay Gateway Edition

Version 3.4.6

Released March 17, 2017

Table of Contents

Changes and Modifications

The table below lists changes made to the WebLINK Developer API Guide:

Version	Changes/Modifications	Pages
1.0.0	Document launch.	All
3.0.0	WebLINK3.0 topics added	All
3.0.1	Corrected PCI warning in section 2.6.4	p. 12
3.1.0	Updated input & output parameter descriptions	All
3.2.0	Updated with Cancel button	pp. 9-10
3.2.1	Added Payment Type Explanation	
3.2.2	Added r_LastFour response field	p. 18
3.2.3	Updated appendix with flow charts and use cases. Added Interfacing to WebLINK chapter and code samples.	
3.2.4	Added details about the MSR applet	pp. 12-13
3.2.5	Verbiage changes	p. 6
3.2.6	Added details about ReturnURL, SessionTimeout and RespondWithLastFailureOnly fields.	pp. 12-13
3.2.7	Added details about UseServerSidePostBacks field.	p. 13
3.2.8	Various corrections.	
3.4.1	Document rewrite. Update list of request and response values. Update template information.	All
3.4.2	Added additional tokenization and wallet information.	16,17,23, 29-32
3.4.3	Added Security warning for Click jack vulnerability	8
3.4.4	Added PurchaseToken Usage documentation	9, 19 & 23
3.4.5	Corrected Sample requests	All
3.4.6	Removed DCP Mode references	All

Chapter 1. Overview

BridgePay's WebLINK interface is a flexible, integrated payment solution for e-Commerce transaction processing. WebLINK accepts transaction request data through an HTTP POST from the merchant website.

WebLINK creates a payment page for you in response to an HTTP POST from a web *<form>* and hosts it on the BridgePay servers. You can customize the payment page to match your website's branding and data collection needs.

Use WebLINK to ensure the merchant is outside of PCI scope and to get up and running quickly. The available template customization features provide enough flexibility for your desired user experience.

WebLINK also generates and displays a receipt for the transaction within the WebLINK interface. After processing the transaction, WebLINK returns transaction details and customer information to the merchant.

	WebLINK
Use Case	Web payment form.
PCI Scope	Merchant is outside PCI scope.
User Interface	Hosted by WebLINK.
Customization	Default payment form. Highly customizable payment form templates available.
Integration Effort	Simple integration with minimal configuration.

In addition, WebLINK can be configured to send receipt emails to customers and transaction confirmation emails to merchants. BridgePay provides standard templates for all pages and emails, but you may create and save your own custom templates using the BridgePay **Merchant Boarding Portal**.

**** Security Warning** – WebLink does not support any iframe integration. Bridgepay has implemented anti-framing methods to ensure that security is in place to prevent vulnerabilities such as click jacking.

Chapter 2. Integration

To use WebLINK, create an HTML form that POSTs to the WebLINK URL. Set the parameters in the form to control and customize behavior. WebLINK generates a payment form for the customer. When the customer completes the payment, WebLINK displays a receipt and sends the results to a specified URL.

See the **Flow Chart** on page 42 for an overview of WebLINK logic. The following sections describe how to use WebLINK in an integration.

2.1. Creating the Form

The first step is to create the form that submits to WebLINK. The *<form>* element must specify a method of post and an action pointing to the WebLINK URL:

https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx

The minimum fields you need to submit on the form are:

- **mode** – This value must be PaymentForm.
- **login** – Your MyBridgePay login.
- **password** or **passwordEncrypted** – Your MyBridgePay password or encrypted password value retrieved from the UAT environment.
- **purchaseToken** – A previously acquired GUID from the BridgeComm **ActionService** (see the BridgeComm v2.3.4 or later documentation for details on acquiring a purchase token).

Include data fields as *<input>* elements in the form. Specify the name of the field in the name attribute and the value in the value attribute. Specify a type attribute value of hidden to prevent the value from being displayed on your HTML page, or use a standard HTML input type to allow customers to fill in values.

The following example configures WebLINK in the simplest format possible. WebLINK creates a hosted payment form, collects all customer and payment data, and handles the receipt.

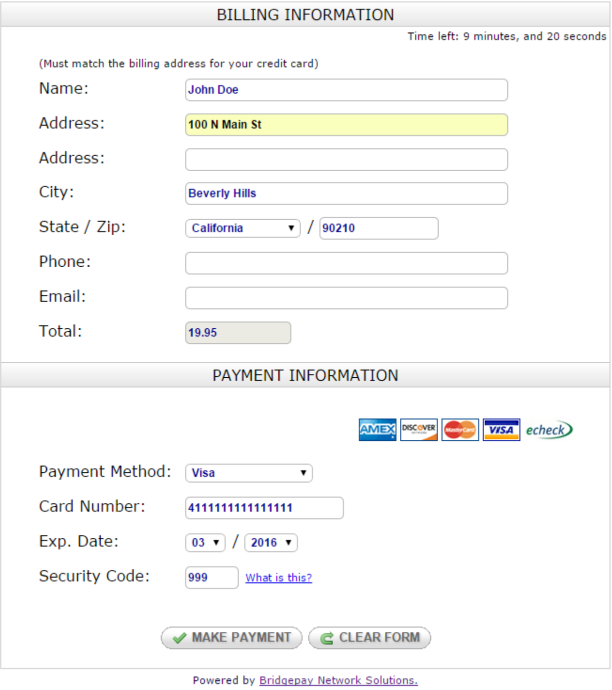
```
<form method=post action="https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx">
  <input type="hidden" name="mode" value="PaymentForm" />
  <input type="hidden" name="login" value="YOUR_MYBRIDGEPAY_LOGIN" />
  <input type="hidden" name="password" value="YOUR_MYBRIDGEPAY_PASSWORD" />
  <input type="submit" value="Submit payment" />
</form>
```

Alternatively, you may use a Purchase Token element instead of your login and password as follows:

```
<form method=post action="https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx">
  <input type="hidden" name="mode" value="PaymentForm" />
  <input type="hidden" name="purchasetoken" value="PREVIOUSLY_ACQUIRED_PURCHASE_TOKEN" />
  <input type="submit" value="Submit payment" />
</form>
```

The remainder of this document will describe messages using the login/password authentication method. However, in all examples, you can replace the login and password elements with a purchase token element if you have acquired a purchase token beforehand.

The following shows an example of the default payment page:



BILLING INFORMATION

Time left: 9 minutes, and 20 seconds

(Must match the billing address for your credit card)

Name: John Doe

Address: 100 N Main St

Address:

City: Beverly Hills

State / Zip: California / 90210

Phone:

Email:

Total: 19.95

PAYMENT INFORMATION

Payment Method: Visa

Card Number: 4111111111111111

Exp. Date: 03 / 2016

Security Code: 999 What is this?

Powered by Bridgepay Network Solutions.

2.2. Sending Additional Data

In addition to the minimum fields, you can send additional data regarding the transaction to the form. Any additional data that you send, such as total amount or billing information, is pre-filled when WebLINK displays the payment form.

The following example pre-fills an amount of $20.00 on the payment form:

```
<form method=post action="https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx">
  <input type="hidden" name="mode" value="PaymentForm" />
  <input type="hidden" name="login" value="YOUR_MYBRIDGEPAY_LOGIN" />
  <input type="hidden" name="password" value="YOUR_MYBRIDGEPAY_PASSWORD" />
  <input type="hidden" name="TotalAmt" value="20.00" />
  <input type="submit" value="Submit payment" />
</form>
```

See **Request Fields** starting on page 23 for a complete list.

2.3. Handling Responses

When operating WebLINK, applications typically need to know when/if a transaction completed successfully in order to record that in the merchant's system. WebLINK provides a few parameters that can be used for this purpose:

- **ReceiptURL** – If provided, WebLINK POSTs all transaction data and re-direct the user's browser to this URL after a successful transaction. If this parameter is not provided, WebLINK simply displays the standard receipt page by default. See **ReceiptURL Sample** on page 44-45 for an example of this page in ASP.
- **FailURL** – If provided, WebLINK POSTs all transaction data and re-directs the user's browser to this URL after a declined transaction. If this parameter is not provided, WebLINK simply displays the payment form again with the error message displayed.
- **CompleteURL** – If provided, WebLINK POSTs all transaction data to this URL. Any output generated from this URL is NOT sent to the user's browser, however. This is the ideal place to do any post transaction work. See **CompleteURL Sample** on pages 43-44 for an example of this page in ASP.

The following example uses WebLINK to display the payment form, collect the customer and payment information, and generate a merchant confirmation email. The receipt, however, displays on the merchant's website, as set in the ReceiptURL value.

```
<form method=post action="https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx">
  <input type="hidden" name="mode" value="PaymentForm" />
  <input type="hidden" name="login" value="YOUR_MYBRIDGEPAY_LOGIN" />
  <input type="hidden" name="password" value="YOUR_MYBRIDGEPAY_PASSWORD" />
  <input type="hidden" name="TotalAmt" value="23.89" />
  <input type="hidden" name="ReceiptURL" value="http://www.somehost.com/receipt.aspx" />
  <input type="hidden" name="MerchantEmail" value="owner@somehost.com" />
  <input type="hidden" name="EmailFrom" value="noreply@somehost.com" />
  <input type="submit" value="Submit payment" />
</form>
```

Once this data is posted to WebLINK, the transaction is processed and an XML response is immediately returned to the caller. The response parameters are contained in the <Data> element.

Every parameter received in the POST message is echoed back to the calling application in the response XML.

The response fields from the gateway are also included in the XML. Response fields are prefixed by r_, for example: r_Result.

The format of the xml is:

```
<Data>
    <Login>loginName</Login>
    ...other posted fields
    <r_Result>0</r_Result>
    ...other result fields
</Data>
```

For each of transaction, address verification, and card verification results, WebLINK returns two fields: a response code (r_Result, r_AvsResult, r_CvResult) and a text description of the response (r_RespMsg or r_Message, r_AvsResultText, r_CvResultTest). Always use the response codes (r_Result, r_AvsResult, r_CvResult) to verify the results. The text description values cannot be guaranteed to be consistent.

See **Response Fields** starting on page 23 for a complete list.

The merchant application should use the data in the response field to display transaction results to the user and to perform any updates to the merchant's database that need to be performed.

2.4. Styling the Payment Form

You can customize the appearance of the default payment form displayed by WebLINK by including style fields in your request.

The following example sets the page background to dark gray, the form background to light gray, and the font to Times New Roman:

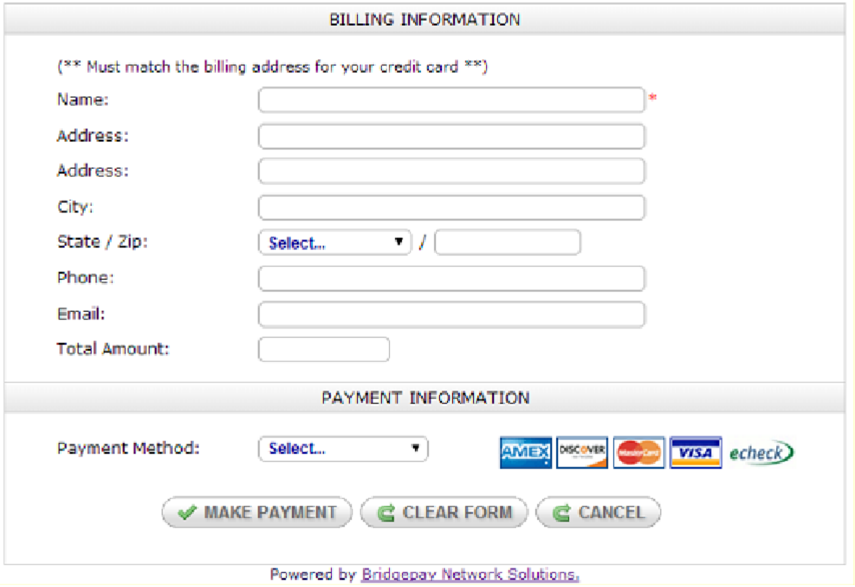
```
<form method=post action="https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx">
   <input type="hidden" name="mode" value="PaymentForm" />
   <input type="hidden" name="login" value="YOUR_MYBRIDGEPAY_LOGIN" />
   <input type="hidden" name="password" value="YOUR_MYBRIDGEPAY_PASSWORD" />
   <input type="hidden" name="Style_Page_BackgroundColor" value="#A9A9A9" />
   <input type="hidden" name="Style_Form_BackgroundColor" value="#D3D3D3" />
   <input type="hidden" name="Style_FontFamily" value="’Times New Roman’" />
   <input type="submit" value="Submit payment" />
</form>
```

See **Payment Form Style Fields** on page 21 for a complete list of style options.

2.5. Adding a Cancel Button

If you submit a CancelURL value, the payment form will include a cancel button. When customers click the cancel button, they are returned to the specified CancelURL.

The following shows an example of a payment form with a cancel button:



BILLING INFORMATION

(** Must match the billing address for your credit card **)

Name: *

Address:

Address:

City:

State / Zip: Select... /

Phone:

Email:

Total Amount:

PAYMENT INFORMATION

Payment Method: Select...

Powered by Bridgepay Network Solutions.

2.6. Using Custom Templates

In addition to customizing the default templates provided by BridgePay, you can also use your own custom payment templates. WebLINK allows you to create custom templates for the payment form, receipts, emails, and timeout errors.

The following example uses a custom payment form template created in the Merchant Boarding Portal. The integrator collects the customer information, leaving WebLINK to collect payment information and display the receipt.

The element `<input type="hidden" name="PaymentFormTID" value="1" />` tells WebLINK to use the custom payment form number one.

```
<form method=post action="https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx">
  <input type="hidden" name="mode" value="PaymentForm" />
  <input type="hidden" name="login" value="YOUR_MYBRIDGEPAY_LOGIN" />
  <input type="hidden" name="password" value="YOUR_MYBRIDGEPAY_PASSWORD" />
  <input type="hidden" name="TotalAmt" value="23.89" />
  <input type="hidden" name="PaymentFormTID" value="1" />
  <input type="text" name="BillToName" value="John Doe" />
  <input type="text" name="BillToStreet" value="100 N Main St." />
  <input type="text" name="BillToCity" value="Beverly Hills" />
  <input type="text" name="BillToState" value="CA" />
  <input type="text" name="BillToZip" value="90120" />
  <input type="submit" value="Submit payment" />
</form>
```

Custom templates are set in the Merchant Boarding Portal. See **Templates** on page 26 for complete instructions on creating and editing templates.

2.7. Split Transaction Processing

A split transaction is a transaction that processes two or more individual transactions in one POST to WebLINK. Each transaction uses the same payment information, but can differ in the amounts and which gateway account the transaction is processed against (i.e., which MID is used). If any transaction in the group fails, processing stops and any prior approved transactions in the group are voided or reversed, depending on processor implementation.

In order to process a split transaction, you must provide a TransSequenceNo of “1” for the first transaction. Subsequent transactions are then indicated by passing in the additional TransSequenceNo, TotalAmt, Login, Password, and optionally, Description fields. These additional fields must be suffixed with an incremental index number. The fields for the second transaction will be named TransSequenceNo1, TotalAmt1, Login1, Password1 and Description1.

The following example shows how to use split transaction processing.

```
  <form method=post action="https://www.bridgepaynetsecuretest.com/WebLink3/WebLink.aspx">
 <input type="hidden" name="Mode" value="PaymentForm" />
 <input type="hidden" name="ReceiptURL" value="https://somesite.com/ShowReceipt.php" />
 <!-- First transaction details -->
 <input type="hidden" name="login" value="YOUR_MYBRIDGEPAY_LOGIN" />

 <input type="hidden" name="password" value="YOUR_MYBRIDGEPAY_PASSWORD" />
 <input type="hidden" name="TotalAmt" value="23.89" />
 <input type="hidden" name="TranSequenceNo" value="1" />
 <input type="hidden" name="Description" value="Water bill" />
 <!-- Second transaction details -->
 <input type="hidden" name="login1" value="usernext" />
 <input type="hidden" name="password1" value="passwordnext" />
 <input type="hidden" name="TotalAmt1" value="33.22" />
 <input type="hidden" name="TranSequenceNo1" value="2" />
 <input type="hidden" name="Description1" value="Trash bill" />
 <!-- Third transaction details -->
 <input type="hidden" name="login2" value="user3" />
 <input type="hidden" name="password2" value="password3" />
 <input type="hidden" name="TotalAmt2" value="2.50" />
 <input type="hidden" name="TranSequenceNo2" value="3" />
 <input type="hidden" name="Description2" value="Convenience fee" />
 <input type="text" name="BillToName" value="John Doe" />
 <input type="text" name="BillToStreet" value="100 N Main St." />
 <input type="text" name="BillToCity" value="Beverly Hills" />
 <input type="text" name="BillToState" value="CA" />
 <input type="text" name="BillToZip" value="90120" />
 <input type="text" name="CardNumber" value="4111111111111111" />
 <input type="text" name="CardExpMonth" value="04" />
 <input type="text" name="CardExpYear" value="2013" />
 <input type="hidden" name="CompleteURL" value="http://www.yourhost.com/receipt.aspx" />
 <input type="hidden" name="FailURL" value="http://www.yourhost.com/fail.aspx" />
 <input type="submit" value="Submit payment" />
</form>
```

2.8. Tokenization

Weblink supports the ability to tokenize a card number and re-use it in place of the actual card number in subsequent transactions.

2.8.1. Basic use

Every transaction performed results in a token being returned to the caller. This token can then be used in subsequent transactions in place of the actual card number. You will still need to provide expiration date and CV number as necessary.

2.9. Wallet

WebLINK supports storing card holder payment information for use in subsequent transactions. This functionality expands upon tokenization by also securely storing additional card holder data such as expiration date, card holder name and AVS data.

2.9.1. Requirements

In order to enable the wallet functionality, you must obtain a wallet site identifier. This is used to logically group your wallets for administration purposes. Please work with your Bridgepay certification specialist to obtain a wallet site identifier.

2.9.2. Basic use

Provide the wallet site id you obtained from your Bridgepay integration specialist and enable the wallet via the EnableWallet request field. The payment form will then display an option for the user to save their current payment information for later use. If they select this option and the transaction is processed successfully, a wallet key and wallet token are returned to the caller.

These two values are then used on subsequent requests to WebLINK to enable the end user to use their stored payment information.

Chapter 3. Request Fields

WebLINK's standard request fields include customer information, payment information, formatting parameters, and fields that govern WebLINK's behavior.

> ⚡ Request field values for form style and application behavior take precedence over the default **Templates and Settings** configured in the Merchant Boarding Portal.

The following tables contains field descriptions. Unless otherwise stated, all fields are optional.

> ⓘ Varchar indicates variable length character string which can include **alphanumeric (0-9, a-z, A-Z and hyphens)** unless otherwise specified.
>
> ⓘ Integer indicates **numeric with maximum length of 9 digits** unless otherwise specified.

3.1. General Purpose Request Fields

The following fields set credentials and basic processing options:

Field	Data Type (Length)	Description
Login	Varchar(25)	Required unless using a **Purchase Token.** The MyBridgePay merchant login that you wish to process the payment against.
Password	Varchar(75)	Either **Password** or **PasswordEncrypted** is required unless using a **Purchase Token**. The password associated with the specified Login.
PasswordEncrypted	Varchar(255)	Either **Password** or **PasswordEncrypted** is required unless using a **Purchase Token**. An encrypted password value available via the test form in the UAT environment by clicking the **Encrypt** button to the right of the **Password** field.
PurchaseToken	Varchar(36)	A previously acquired GUID from BridgeComm's **ActionService**.

Field	Data Type (Length)	Description
Mode	Enumeration	Required. Valid values are: **PaymentForm | TokenizationForm** **PaymentForm:** displays a form to collect payment data for processing a transaction. **TokenizationForm:** displays a form to capture card info, returning a token.
TransType	Varchar(10)	The transaction type to perform. Valid values are: **Sale | Auth** **Sale:** performs a sale transaction. **Auth:** performs an authorization only transaction; the amount is authorized but not collected from the customer. (The default value is Sale.)

3.2. Credit Transaction Detail Request Fields

The fields in this table describe the credit transaction being submitted

Field	Data Type (Length)	Description
TotalAmt	Decimal(10,2)	The total amount of the transaction. Leaving this blank allows customers to enter their own amount into the standard payment form. The decimal should be explicitly entered with the amount (e.g., 5.83; 0.99).
CardTypeName	Varchar(10)	Card type. Valid values are: **Visa, MasterCard, Discover, Amex, eCheck**
CardNumber	Varchar(16)	Credit card number.
CardExpMonth	Char(2)	Two digit credit card expiration month. (e.g., 02 for February).

Field	Data Type (Length)	Description
CardExpYear	Varchar(4)	Two or four digit credit card expiration year (e.g., 2013 or 13).
CVNum	Varchar(4)	Three or four digit credit card security code (a.k.a., CVV, CV2).
Description	Varchar(255)	The description of the transaction.
FullName	Varchar(150)	Full name of the cardholder.
BillToStreet	Varchar(128)	The cardholder’s street address.
BillToCity	Varchar(50)	The cardholder’s city.
BillToState	Char(2)	The cardholder’s state.
BillToZip	Varchar(15)	The cardholder’s ZIP code. (5 or 5+4 and if 5+4, NNNNN or NNNNNNNNN).
Email	Varchar(100)	The email address of the cardholder. If this field is included, an email receipt is sent to this address after a successful transaction.
MerchantEmail	Varchar(100)	The email address of the merchant. If included, an email receipt is sent to this address after a successful transaction.
EmailFrom	Varchar(100)	Email address WebLINK specifies as the "from" address for customer and merchant emails.
InvoiceNum	Varchar(24)	The invoice number to associate with the transaction.
PONum	Varchar(24)	Purchase order number to associate with the transaction.
Token	Varchar(40)	The token to process the transaction against.
TransSequenceNo	Integer	Required only for **split transactions**. The transaction sequence number for this transaction.

3.3. Behavior Request Fields

These fields influence how the WebLINK application processes.

Field	Data Type (Length)	Description
ReceiptURL	Varchar(255)	If the transaction submits *successfully*, WebLINK performs an HTTP POST to this URL for a receipt or transaction confirmation. Leave this field blank to allow WebLINK to display the receipt automatically. When left blank, WEBLINK displays the receipt in addition to other behaviors, such as the CompleteURL behavior discussed below.
ReturnReceiptUrl	Varchar(255)	If you choose to have WebLINK handle the receipt and do not provide a value for **ReceiptURL**, WebLINK provides a link on the receipt page labeled “Return to [Merchant Name]” that will redirect the browser to the **ReturnReceiptUrl**.
CompleteURL	Varchar(255)	After a transaction is processed, regardless of its result status, the results are POSTed to CompleteURL. ⓘ If the input parameters passed to WebLINK are configured for WebLINK to display the receipt, WebLINK displays the receipt *and* POSTs the transaction result to the CompleteURL ⓘ This URL can post transaction data back to a server inside your firewall to confirm legitimate transactions. ⓘ The results of this POST, if any, are *not* rendered back to the browser for display to the user. Other receipt behaviors will need to be passed to handle receipt display.

Field	Data Type (Length)	Description
FailURL	Varchar(255)	WebLINK POSTs to this URL if the payment form submission is *unsuccessful*. All form fields, including transaction result, is POSTed to this URL. ⓘ Leaving this blank causes WebLINK to show the payment form again with any response from the payment gateway displayed on the form.
CancelURL	Varchar(255)	If provided, a cancel button is rendered on the payment form that, when clicked, redirects to the CancelURL. ⓘ Adding a space and then "confirm" causes a confirmation dialog to appear before the end user is re-directed. E.g., "http://www.domain.com/cancelTransaction.php confirm".
ReturnURL	Varchar(255)	The URL that users are returned to when their session times out.
ReceiptLogoUrl	Varchar(255)	The URL to the logo image to display on receipts. Provide this value to display a logo image on receipts that is different from the values configured in the Merchant Boarding Portal.
PaymentServerType	Enumeration	Type of payment server to interface with. Valid values: **BPN** **BPN:** uses the BridgePay payment server. For live transactions.
PaymentTypes	Varchar(50)	The payment types to display on the payment form. This also enables the Card Type to be displayed in the drop down box. This is a comma delimited string of the following possible values: **AMEX, Discover, Visa, Mastercard, eCheck** The default value is: AMEX, Discover, Visa, Mastercard, eCheck ⓘ The values are case sensitive.

Field	Data Type (Length)	Description
SessionTimeout	Integer	The amount of time in minutes before the user's session times out. Min: 1, Default: 10, Max: 30
RespondWithLastFailureOnly	Varchar(1)	Provide a non-empty string in this field to indicate to WebLINK that a response of a declined transaction to the CompleteURL parameter URL should only occur after the user's session times out and should only include the last failure if there were more than one. Successful transactions will always be POSTed back to CompleteURL immediately regardless of this setting.
UseServerSidePostBacks	Varchar(1)	Provide a non-empty string in this field to indicate to WebLINK that it should POST results to ReceiptURL and FailURL from the server instead of a client-side re-direct. Server-side posting is more secure than client-side, but can only be used with publicly accessible URLs. Default value is [null].

3.4. Template Selection Request Fields

The following fields allow you to set custom templates to use for the payment form, receipts, and email. If you do not submit values, WebLINK uses the default forms.

Field	Data Type (Length)	Description
PaymentFormTID	Integer	The template ID specifying which payment form to display. Applicable to the normal payment form, tokenized payment form and the token creation form.
ReceiptTID	Integer	The template ID specifying which receipt form to display.
CustomerEmailTID	Integer	The template ID specifying which customer email to send.
MerchantEmailTID	Integer	The template ID specifying which merchant email to display.

3.5. Payment Form Style Fields

The following fields can be used to customize the look and feel of the WebLINK payment collection form.

Field	Data Type (Length)	Description
Style_Page_BackgroundImage	Varchar(255)	The URL to an image file to be included as the background image for the payment form.
Style_Page_BackgroundColor	Varchar(7)	The color to use for the page background color. The default value is “#FFF”.
Style_Form_BackgroundColor	Varchar(7)	The color to use for the payment form background color. The default value is “#FFF”.
Style_FontFamily	Varchar(255)	The font family to use on the payment form. One or more font names, separated by commas. The default value is “Verdana,Arial,Helvetica,sans-serif”.
Style_Heading_FontColor	Varchar(7)	The color to use for heading text on the payment form. The default value is “#000”.
Style_Heading_FontSize	Integer	The size of the font to display for heading text on the payment form. The default value is “13”.
Style_Heading_BackgroundColor	Varchar(7)	The background color for the heading sections of the payment form.
Style_Instruction_FontColor	Varchar(7)	The color to use for instruction text on the payment form. The default value is “#000”.
Style_Instruction_FontSize	Integer	The size of the font to display for instruction text on the payment form. The default value is “11”.
Style_Label_FontColor	Varchar(7)	The color to use for the field label text on the payment form. The default value is “#000”.
Style_Label_FontSize	Integer	The size of the font to display for field label text on the payment form. The default size is “10”.
Style_Field_FontColor	Varchar(7)	The color to use for field input text on the payment form. The default value is “#000”.
Style_Field_FontSize	Integer	The size of the font to display for field input text on the payment form. The default size is “12”.

3.6. Wallet Request Fields

The following fields are used when you wish to take advantage of the wallet functionality.

Field	Data Type (Length)	Description
EnableWallet	Varchar(1)	Provide a non-empty string in this field to enable wallet functionality.
SiteId	Guid	The id of the site to create the wallet under. This value should only be provided when creating a new wallet. Leave this field blank if you are using a previously created wallet.
WalletToken	Varchar(500)	The Wallet Token of the wallet you wish to use for the transaction. Do not provide a value here if you are creating a new wallet.
WalletKey	Varchar(500)	The Wallet Token of the wallet you wish to use for the transaction Do not provide a value here if you are creating a new wallet.

Chapter 4. Response Fields

All request fields are echoed back in the response messages sent by WebLINK. The transaction results from the host are also included, prefixed by r_.

These fields are available for use in email and receipt templates, and when specified, these fields post to **FailURL**, **CompleteURL**, and **ReceiptURL**.

> Some fields are only available after initiating the transaction (e.g., AuthCode).

The following table contains field descriptions of the host response fields:

Field	Data Type (Length)	Description
r_AuthCode	Varchar(50)	The transaction result code from the payment processor. Format varies by processor.
r_HostCode	Varchar(30)	A number that uniquely identifies the transaction in the payment processor. Format varies by processor. Not all payment processors return this value.
r_Message	Varchar(255)	A formatted response message concerning the processed transaction. This value is typically **APPROVAL** for approved transactions or an error message for declined transactions. **Never** use this value when programmatically validating a transaction's result; please see the **r_Result**.
r_Result	Varchar(50)	Result code that signifies the result of the transaction. When programmatically validating a transaction's result, **always** use this value instead of any response message describing the result.
r_Id	Integer	Number that uniquely identifies the transaction in the payment gateway. (E.g., 76543, 245683.)
r_Token	Integer	The tokenized card number returned from the payment server. This can be used in subsequent transactions in place of the CardNumber field by sending the token in the Token field and leaving the CardNumber field blank.

Field	Data Type (Length)	Description
r_CardType	Varchar(50)	The card type name of the card used. This field is only populated when the CardNumber field is used.
r_LastFour	Varchar(4)	The last four digits of the card number used in the transaction.
r_AuthorizedAmount	Decimal	The actual amount authorized to the card. This amount will differ from the requested amount for partial approvals.
r_WalletToken	Varchar(500)	The wallet token of the newly created wallet if applicable.
r_WalletKey	Varchar(500)	The wallet key of the newly created wallet if applicable.

Chapter 5. Templates and Settings

The following sections discuss how to use the BridgePay Merchant Boarding Portal to configure templates and default settings. The Merchant Boarding Portal is available at:

https://www.mybridgepaytest.com/

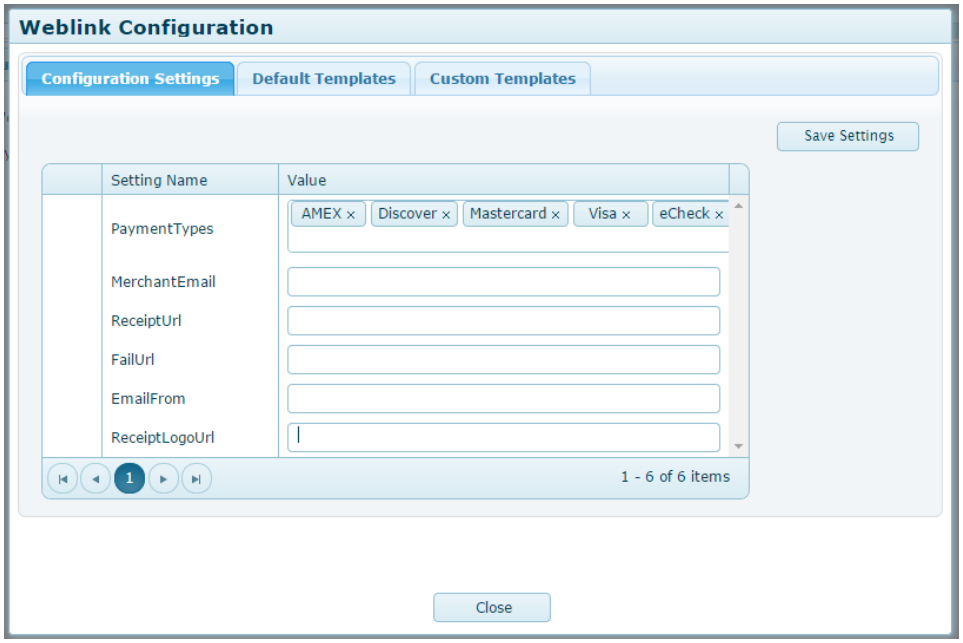
5.1 Settings

The application settings available for configuration are:

- **PaymentTypes**
- **MerchantEmail**
- **ReceiptURL**
- **FailURL**
- **EmailFrom**
- **ReceiptLogoURL**

For descriptions of these fields, see **Request Fields** starting on page 15.

> Remember that you can also set the values for these fields on a case-by-case basis in the form post. Values in the form posts take precedence over values set in the Merchant Boarding Portal.



Setting Name	Value
PaymentTypes	AMEX × Discover × Mastercard × Visa × eCheck ×
MerchantEmail	
ReceiptUrl	
FailUrl	
EmailFrom	
ReceiptLogoUrl	

5.2 Templates

WebLINK uses these types of templates:

- **CustomerEmail** – Customizes the display of the transaction receipt email sent to customer.
- **MerchantEmail** – Customizes the display of the transaction confirmation email sent to the merchant.
- **PaymentForm** – Configures the settings and display of the payment form.
- **PaymentFormSplit** – Configures the settings and display of the payment form for split transactions.
- **TokenziationForm** – Configures the settings and display of the tokenization form.
- **Receipt** – Configures the settings and display of the transaction receipt.
- **ReceiptSplit** – Configures the settings and display of the transaction receipt forsplit transactions.
- **SessionTimeout** – Configures the settings and display of the page when a session timeout occurs.

There are two categories of templates: **merchant** and **custom**. WebLINK invokes merchant templates by default if you don't pass a custom template ID in the request data. You may also create a custom template and invoke it on-demand by passing the corresponding **Template ID (TID)** when posting to WebLINK.

To customize a merchant template, paste your formatted code into the **HTML** field for the specified template and click **Save**. To create a custom template, select the template type from the dropdown menu and click **Create**. A new template entry appears with a new TID. Paste your custom template code into the **HTML** field and click **Save**.

You are responsible for pasting in the proper HTML for the new template. See **Request Fields** on page 23 for a list of custom template ID fields and **Formatting Templates** below for information on formatting the template HTML properly.

5.2.1. Formatting Templates

All templates consist of HTML pages. HTML pages can be customized using various WebLINK-specific placeholder tags. The following sections describe how to use the placeholder tags. See the default templates in the merchant boarding portal for more examples on how to create templates.

5.2.1.1. Parameter Placeholders

WebLINK uses unique placeholders inside of templates to render the payment form, payment receipt, merchant transaction confirmation email, and customer transaction receipt email.

You structure these templates just like typical HTML documents, but you also include specific placeholders that WebLINK uses to assemble the form or email.

Placeholder Format

```
<!--[PlaceholderName]required-->
```

> The **required** keyword is optional. If specified, the customer must complete the field or WebLINK returns a validation error when submitting the form.

If the customer's browser supports JavaScript, there is client-side validation in addition to server-side validation after submitting the payment form. WebLINK automatically generates the necessary JavaScript code for you.

You can use placeholders almost anywhere in an HTML template. Simply use the placeholder inside an HTML tag, such as a <span> and WebLINK displays the placeholder's value in the template.

```
<span><!--[PlaceholderName]--></span>
```

You can also use a placeholder as the value attribute of an <input> element. In this case, WebLINK pre-fills the input field with the value of the placeholder when rendering the template.

```
<input type="text" name="PlaceholderName" value="<!--[PlaceholderName]required-->" />
```

> The <input> element's name attribute must match the name of the placeholder.

5.2.1.1.1.Required Placeholders

These placeholders must be in every template in order for WebLINK to function properly:

Placeholder	Description
<!-- %%STYLES%% -->	This must reside inside the <head> tag of the template.
<!-- %%HEAD_SCRIPT%% -->	This must reside in the <head> tag of the template.
<!--%%BODY_SCRIPT%% -->	This must reside before the closing </body> tag.

5.2.1.1.2.Security Logo Placeholders

These optional placeholders display security logos:

Placeholder	Description
<!—{{ thawte }} -->	This is the placeholder for the Secured by Thawte logo and must reside within the <body> of the document.

Placeholder	Description
<!—{{ align }} -->	This is the placeholder for the A-lign PCI DSS logo and must reside within the *<body>* of the document.

5.2.1.1.3.Reserved Placeholders

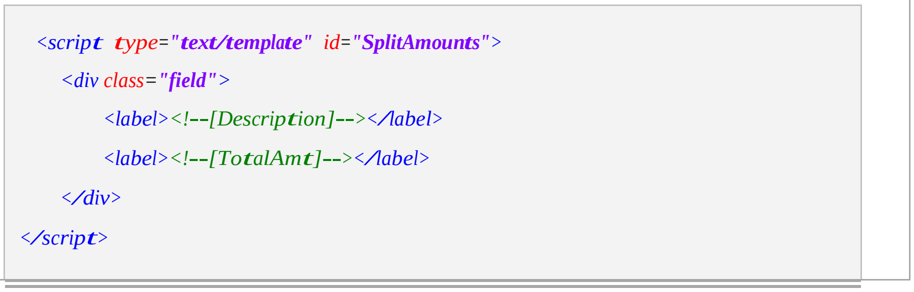
All **Request Fields** are reserved placeholders. For example, if you put **<!--[InvoiceNum]-->** in the HTML template for the payment form and submitted a value of **123456** in the text box, the payment server receives 123456 as the invoice number for the transaction.

5.2.1.1.4.Special Reserved Placeholders

The following table contains special reserved placeholders. See **Parameter Placeholders** on page 36 for an example of placeholder usage.

Placeholder	Description
TotalAmt	Renders as an input tag for the **TotalAmt** field in the payment form and plaintext for emails and receipts. (E.g., 3.63.)
BillToState	Renders a select tag for the **BilltoState** field.
CardTypeName	Renders a select tag for the card type.
PaymentTypeImages	Renders *<img>* tags for the available payment types.
CardExpMonth	Renders a select tag for the **CardExpMonth** field.
CardExpYear	Renders a select tag for the **CardExpYear** field. Year range includes the current year + 9 years.
CheckType	Renders a select tag for the **CheckType** field. Possible values include: **Personal | Corporate | Government**
AcctType	Renders a select tag for the **AcctType** field. Possible values include: **Savings | Checking**
ValidationErrors	Renders an unordered list tag and list elements for each server validation error WebLINK encounters.
GenericAcctNum	Displays the masked credit card or checking account number.
TransDate	Displays the date and time of transaction processing.

Placeholder	Description

SplitAmounts	Renders each transaction line item for split payment transactions. This placeholder is only used in the PaymentFormSplit template.
	The following HTML template (from the bottom of the default PaymentFormSplit template) can be customized to alter how each transaction total line item appears:
	
	`<script type="text/template" id="SplitAmounts">`
	`<div class="field">`
	`<label><!--[Description]--></label>`
	`<label><!--[TotalAmt]--></label>`
	`</div>`
	`</script>`

5.2.1.1.5.Error Placeholders

Errors display on the payment form with two reserved placeholder fields, **r_Message** and **ValidationErrors**. r_Message contains a formatted response message from the payment gateway, describing the status of the processed transaction. ValidationErrors contains a comma-delimited list of errors deriving from improper data entry on the initial post to WebLINK or from the end-user submitting the payment form. If you specify a **FailURL**, errors post to the given URL instead of displaying on the payment form.

> ⓘ Validation errors resulting from the end-user's form submission are rare as the forms use client-side validation to handle these types of errors.

5.2.1.1.6.Conditional HTML Placeholders

Conditional HTML placeholders consist of an opening tag and closing tag. Any HTML between the tags is conditionally rendered by WebLINK based on the following rules:

Placeholder	Description
<!—PaymentInformation --> ***HTML code*** **<!-- /PaymentInformation -->**	If a token is POSTed to WebLINK, then WebLINK does not render this HTML. WebLINK assumes that payment information does not need to be collected from the user.
<!—WalletUse --> ***HTML code*** **<!-- /WalletUse -->**	If a WalletToken, WalletKey and EnableWallet are POSTed in to WebLINK, then WebLINK renders this HTML.
<!—WalletSave --> ***HTML code*** **<!-- /WalletSave -->**	If a SiteId and EnableWallet are POSTed to WebLINK, then WebLINK renders this HTML.
<!—CancelPayment --> ***HTML code*** **<!-- /CancelPayment -->**	If CancelUrl is POSTed to WebLINK, then WebLINK renders this HTML.

5.2.1.1.7.Custom Placeholders

You can also create custom placeholders to use in the payment form to gather and pass to the receipt or CompleteURL. Format custom placeholders like WebLINK's reserved placeholders.

For example, if you want to give customers a space to include an optional note concerning their order:

```
<input type="text" name="<!—[AdditionalNotes]-->" value="<!—AdditionalNotes-->">
```

> Since the placeholder was used in the value parameter, any initial value for “AdditionalNotes” POSTed to WebLINK is rendered in the form. If you do not need to pre-fill the input value, you can simply omit the value attribute.

5.2.1.2. Example

The following is an example of the special placeholder for **CardTypeName** to generate an HTML `<select>`, followed by the rendered HTML

Placeholder

```
<div class="field">
        <label>Payment Method:</label>
        <!--[CardTypeName]required-->
        <div id="card-images"><!--[PaymentTypeImages]--></div>
</div>
```

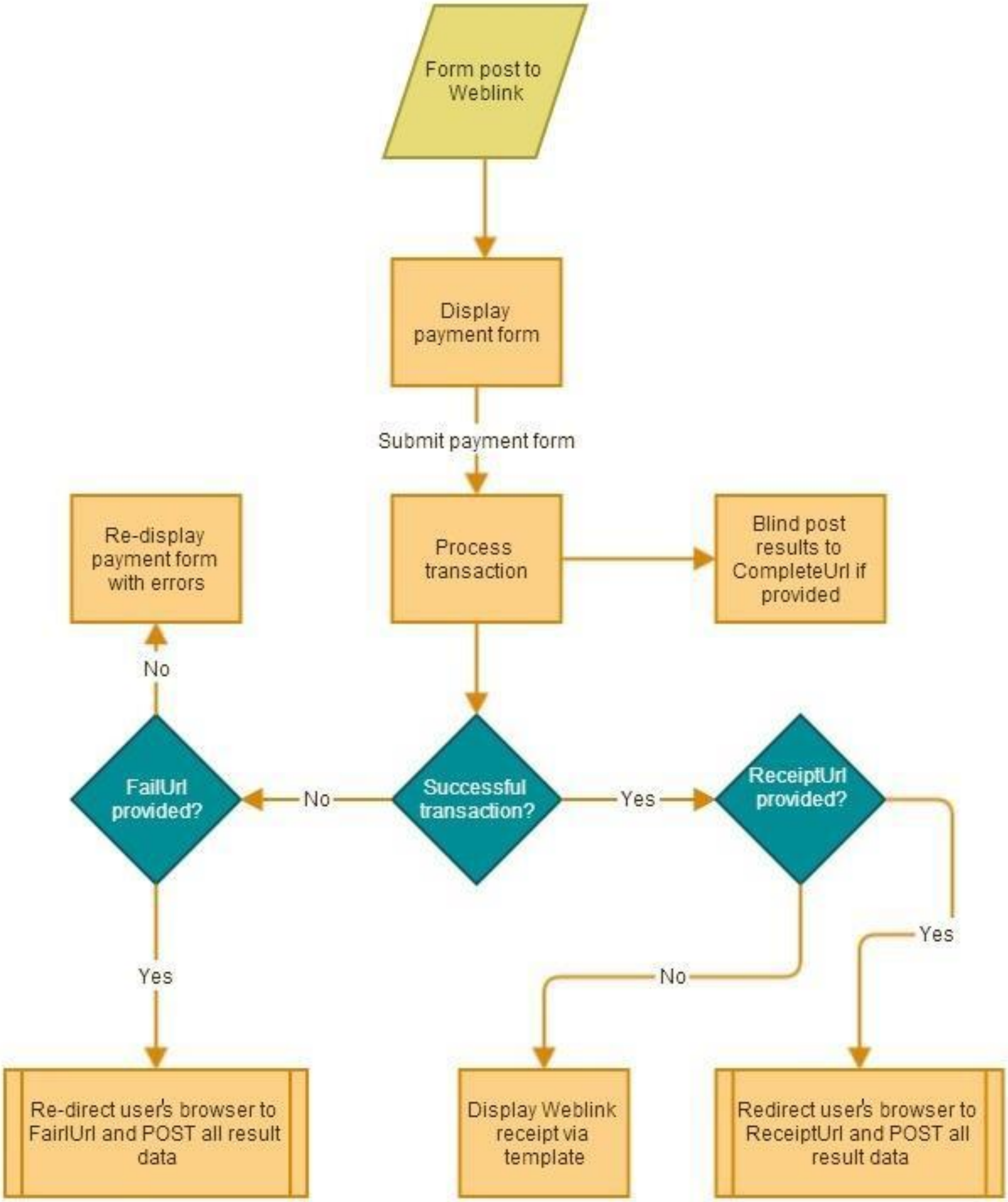
Rendered HTML

```
<div class="field">
        <label>Payment Method:</label>
        <select name="CardTypeName" class="field">
                <option value="">Select...</option>
                <option value="AMEX" st="cc">American Express</option>
                <option value="Discover" st="cc">Discover</option>
                <option  value="Mastercard" st="cc">Mastercard</option>
                <option  value="Visa"  st="cc">Visa</option>
                <option value="eCheck"  st="check">Electronic Check</option>
        </select>

        <div id="card-images">
                <img  src="images/AMEX.gif"  class="payment_method_logo"><img  src="images/Discover.gif"
class="payment_method_logo"><img  src="images/Mastercard.gif"  class="payment_method_logo"><img src="images/Visa.gif"
class="payment_method_logo"><img  src="images/eCheck.gif"  class="payment_method_logo">
        </div>
</div>
```

A. Appendix

A.1. Flow Chart


Form post to Weblink
Display payment form
Submit payment form
Process transaction
Blind post results to CompleteUrl if provided
Re-display payment form with errors
No
FailUrl provided?
No
Successful transaction?
Yes
ReceiptUrl provided?
Yes
No
Yes
Re-direct user's browser to FairlUrl and POST all result data
Display Weblink receipt via template
Redirect user's browser to ReceiptUrl and POST all result data

A.2. Code Samples

A.2.1. CompleteURL Sample

The following ASP code is an example of a CompleteURL page. This page receives a POST containing the result of the transaction, regardless of the transaction result. This page should display the results to the customer and perform any needed updates to the merchant's system.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WeblinkSample
{
    public partial class WeblinkCompleted : System.Web.UI.Page
    {
        // This page is an example of using the CompleteURL parameter. This page will get POSTed to
regardless of the result of the transaction.
        // The output of this page if any, is not rendered back to the customer's browser. protected
        void Page_Load(object sender, EventArgs e)
        {
            var postData = Request.Form;

            // If nothing was POSTed, then there is nothing to do
            if (postData.Count == 0)
            {
                return;
            }

            // Determine if this was an approval or not.
            var isApproval = false;
            int tempResultCode = -1;

            // Anything other than 0 is a failed transaction
            if (int.TryParse(postData["r_Result"], out tempResultCode) &&
                tempResultCode == 0)
            {
                isApproval = true;
            }

            if (isApproval)
            {
                UpdateTransactionDetails(postData);
            }
            else
            {
                LogTransactionFailure(postData);
            }

        }

        private void LogTransactionFailure(System.Collections.Specialized.NameValueCollection postData)
        {
            string message = postData["r_Message"];
            string result = postData["r_Result"];
            string nameOnCard = postData["BillToName"];
            string totalAmount = postData["TotalAmt"];
            string maskedCardNumber = postData["CardNumber"];

            // Todo: Log the transaction in a database, log file, etc
        }

        private void UpdateTransactionDetails(System.Collections.Specialized.NameValueCollection postData)
        {
            // Bridgepay unique transaction identifier
            string transactionId = postData["r_Id"];

            // This assumes you POSTed this "myUniqueIdentifier" field to Weblink during the initial call
            string myUniqueIdentifier = postData["myUniqueIdentifier"];

            // Todo: Update a database with the transaction details
        }
    }
}
```

A.2.2. ReceiptURL Sample

The following ASP code is an example of a ReceiptURL page. This page receives a POST containing the result of a successful transaction in order to display a receipt.

The sample page displays the receipt.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WeblinkReceipt.aspx.cs"
Inherits="WeblinkSample.WeblinkReceipt" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
```

```
    <title>Your receipt</title>
</head>
<body>
    <div>
        <h1><%= Request["MerchantName"] %></h1>
        <h2>Merchandise receipt</h2>
        <table>
            <tr>
                <th><%= Request["Description"] %></th>
                <td><%= Request["TotalAmt"] %></td>
            </tr>
        </table>
        <div><a href="http://www.yourURL.com">Click</a> to return to <%= Request["MerchantName"]
%></div>
    </div>
</body>

</html>
```

The sample code processes the results and performs any needed updates to the merchant’s system.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WeblinkSample
{
    public partial class WeblinkReceipt : System.Web.UI.Page
    {
        // This page is an example of using the ReceiptURL parameter. This page will only be POSTed to for
approved transactions.
        // The output of this page is rendered back to the customer's browser. protected
        void Page_Load(object sender, EventArgs e)
        {
            // You may perform any post processing you wish here such as updating a database with the
transaction results.
        }
    }

}
```