



Commerce Manager

Pass-through Authentication Specification

©2018 Nelnet Business Solutions. All rights reserved.

Nelnet Business Solutions (NBS)
121 S. 13th Street, Suite 301
Lincoln, NE 68508

Copyright Notice

COPYRIGHT©2018 *Nelnet Business Solutions, Inc.* This document is unpublished and the foregoing notice is affixed to protect *Nelnet Business Solutions, Inc.* in the event of advertent publication. All rights reserved. No part of this document may be reproduced in any form, including photocopying or transmission to any computer or organization intranet site without prior consent of *Nelnet Business Solutions, Inc.* The information contained in this document is confidential and proprietary to *Nelnet Business Solutions, Inc.* and may not be used or disclosed except as expressly authorized in writing by *Nelnet Business Solutions, Inc.*

Trademarks

Other product names mentioned in this document may be trademarks or registered trademarks of their respective companies and are hereby acknowledged.

Ownership

This document is the responsibility of *Nelnet Business Solutions, Inc.*

Document Control and History

Version	Date	Author	Comments
2.0	Apr 2012	J. Kray	Edit, reformat original document
2.1	June 2014	A.Bracciano	Edit, add new features
2.2	Jan2015	A.Bracciano	Added Credit Card Last 4 param.
2.3	July 2015	A Bracciano	Updated parameter list table
2.4	Nov 2015	A Bracciano	SHA-256 update,url example

Contents

Introduction	4
Scope	4
About This Document	4
PCI Compliance.....	4
Overview	5
What is Pass-Through Transaction Processing?	5
Private Transaction Processing.....	5
Building a request for Private Transaction Processing – Basic Parameters	6
Building a request for Private Transaction Processing – Optional Parameters.....	6
Example of building pass-through url:.....	7
Appendix	9
Appendix A: Pass-through Parameter Specifications	9
Appendix B: Secure SHA-256 Hash	13
Appendix C: Generating the Time Stamp Parameter.....	14
Appendix D: Optional Feature – Redirect URL	14
Appendix E: Optional Feature – Embedding Commerce Manager in an iframe	18
Appendix F: Optional Feature – Payment Method Flag	19


Introduction

Scope

This document describes the steps necessary to configure the Commerce Manager Pass-through Authentication for an institution. The audience intended for this document is any internal Nelnet team member who is participating in the implementation, configuration or support of the feature.

About This Document

For documentation purposes, any URL and code examples are displayed in a separate font style (`Courier New`) to distinguish from documentation narrative.

The information  icon appears in this document to indicate special notes and instructions.

PCI Compliance

The PCI DSS (Payment Card Industry Data Security Standard) is designed to ensure that all entities storing, processing, or transmitting credit card information are protecting that card holder data in accordance with best practices. *QuikPAY/Commerce Manager* provides institutions and enterprises the ability to outsource online payment processing to an officially recognized and annually audited PCI Level 1 Compliant Service Provider, Nelnet Business Solutions.

Simply using a compliant service provider does not absolve a merchant from all other responsibility pertaining to PCI. For example, local policies and procedures for protecting card holder data must be implemented by all merchants. Also, certain implementations of *QuikPAY* or one of its components may require additional measures be taken at your institution.

For information on PCI policies and procedures specific to your institution, please contact your system administrator or Information Security Officer. If you have additional questions, please contact your NBS Customer Relationship Manager. Additional information may be found at the following web sites:

<http://www.campuscommerce.com>

<http://www.pcisecuritystandards.org>


Overview

What is Pass-Through Transaction Processing?

Pass-through transaction processing allows an institution to process private transactions through Commerce Manager when they have not authenticated the user on their website, that is, the transactions originate from an institution's website that may not require the user to log in.

Private Transaction Processing

Private transactions processed in Commerce Manager require the calculation of a secure hash. Commerce Manager supports a transitive trust pass-through model by using an SHA-256 hash function. Private Transaction Processing maintains transaction integrity because:

-  Commerce Manager will only accept transactions that originated from the institution's web site. Data being passed into Commerce Manager cannot be modified during the session unless specifically allowed. The pass-through transaction request is only valid for 5 minutes after being generated.


Commerce Manager must validate the transaction that is being sent from the source site. This specification for this validation requires the criteria based on keyed SHA-256 hash transaction processing.

Each request is validated on 3 criteria. **A failure of any of the validations will result in a denied request.** The criteria are described below:

- **URL** – The HTTPS request must be sent to the correct URL and contain the required parameter name/value pairs, that have been configured during Commerce Manager Order configuration.
- **Secure SHA-256 Hash** – The value of the hash parameter is compared with a hash generated by Commerce Manager using the other parameter values sent in the HTTPS request and an agreed upon secret key (string). The transaction request will not be allowed to continue if the two values differ. This restriction validates that the request is being sent by a valid party.
- **Time stamp** – The value of the time stamp parameter will be compared with a time stamp generated by Commerce Manager when the HTTPS request is received. The request will not be authenticated if the absolute value of the difference of the two timestamps is greater than 600,000 milliseconds (5 minutes). This time difference restriction will effectively expire the URL so it can't be reused at a later time. See [Appendix C: Generating the Time Stamp Parameter](#) for instructions generating the time stamp.

Building a request for Private Transaction Processing – Basic Parameters

Pass-Through Transaction Processing, in its simplest form, requires only 2 parameters, Order Type and Time Stamp.


 You will need to append these 2 parameters to your key to create the hash digest that Commerce Manager will use to validate the authenticity of the payer.

Hash	Parameter	Parameter Name	Passed to QuikPAY
1	Order Type	orderType	Yes
2	Time Stamp	timestamp	Yes
3	Key	NA	No
NA	Hash	hash	Yes

Order Type – The Order Type you pass into Commerce Manager must correspond to one you configured using the Commerce Manager Administrator> Add New Order. This relationship will ensure that the order’s payment options – including required fields and processors – selected when creating the new order, are applied when a payer is passed in with this Order Type.

Time Stamp – The Time Stamp is a long integer number of the millisecond from the year 1970 (epoch time). See [Appendix C: Generating the Time Stamp Parameter](#) for additional information about generating the Time Stamp.

Key – The Key value that is used along with the Order Type and the Time Stamp to generate the hash digest which will be used to verify the authenticity of the payer being passed into Commerce Manager. Longer, randomly generated keys should be used to maximize security. Key administration options in Commerce Manager include an option to generate a random key.

 When a new order is migrated to the production Commerce Manager environment, a new production key will need to be generated.

Hash – The Hash is a product of the concatenation of the Order Type, Time Stamp and Key being passed through your hash (SHA-256) library/function; if optional parameters are being passed into Commerce Manager, they will need to be hashed as well. See the [Appendix](#) sections for more information.


Building a request for Private Transaction Processing – Optional Parameters

Additional, optional parameters can be passed into Commerce Manager during transaction processing. These parameters can:

- Pre-populate fields for display to the payer in the Commerce Manager screens.

- Be captured/stored in the background of the session and passed back with the transaction details.
- Activate other, optional Commerce Manager features.

If these optional fields are used they will need to be included in the hash digest. When creating the string to hash, they need to be concatenated in order based on the table below. You should only hash the parameters you are sending. If you omit a parameter simply continue building your hash string with the next parameter you are using based on the order of the table below.

 The list of parameters you are hashing needs to be consistent with the parameters that will be passed into Commerce Manager with a transaction.

When additional optional parameters are used, they will need to be appended in the proper hash order to the Order Type, Time Stamp and Key to create the hash digest that Commerce Manager will use to validate the authenticity of the payer. See [Appendix A: Pass-through Parameter Specifications](#) for detailed information on the parameter list.

Example of building pass-through url:

https://uatquikpayasp.com/xyzschool/commerce_manager/payer.do?

Parameters to be passed in

orderType=Yourordertype

orderNumber=17

orderName=JoeSmith

orderDescription=Test

amount=100

redirectUrl= <https://abc.xyz.edu/Yourname/Confirmation.com>

redirectUrlParameters=transactionType,transactionStatus,transactionId,transactionTotal Amount,transactionDate,transactionAccountType,transactionResultCode,transactionResultMessage,orderNumber,payerFullName

retriesAllowed=1

timestamp= 1394219109822

key = key

HASH Creation

Yourordertype17JoeSmithTest100https://abc.xyz.edu/Yourname/Confirmation.comtransactionType,transactionStatus,transactionId,transactionTotalAmount,transactionDate,transactionAccountType,transactionResultCode,transactionResultMessage,orderNumber,payerFullName11394219109822key

hash= 3e3ce3d9ca60ed3a87e09b369e16adecb7115287094e04f1786205b0b47c8b06

FINAL URL

https://uatquikpayasp.com/xyzschool/commerce_manager/payer.do?orderType=Yourordertype&orderNumber=17&orderName=JoeSmith&orderDescription=Test&amount=100&redirectUrl=https://abc.xyz.edu/Yourname/Confirmation.com&redirectUrlParameters=transactionType,transactionStatus,transactionId,transactionTotalAmount,transactionDate,transactionAccountType,transactionResultCode,transactionResultMessage,orderNumber,payerFullName&retriesAllowed=1×tamp=1394219109822&hash=3e3ce3d9ca60ed3a87e09b369e16adecb7115287094e04f1786205b0b47c8b06

Appendix

Appendix A: Pass-through Parameter Specifications

Hash Ord.	Parameter	Parameter Name	Passed to QuikPAY	Data Type	Max Length
1	Order Type	orderType	Yes	String	32
2	Order Number	orderNumber	Optional	String	32
3	Order Name	orderName	Optional	String	32
4	Order Description	orderDescription	Optional	String	32
5	Amount	amount	Optional	Cents, no decimal	12
6	Order Fee	orderFee	Optional	Cents, no decimal	12
7	Current Amount Due	currentAmountDue	Optional	Cents, no decimal	12
8	Balance	balance	Optional	Cents, no decimal	12
9	Current Balance	currentBalance	Optional	Cents, no decimal	12
10	Due Date	dueDate	Optional	Date, yyyyymmdd	8
11	User Choice 1	userChoice1	Optional	String	50
12	User Choice 2	userChoice2	Optional	String	50
13	User Choice 3	userChoice3	Optional	String	50
14	User Choice 4	userChoice4	Optional	String	50
15	User Choice 5	userChoice5	Optional	String	50
16	User Choice 6	userChoice6	Optional	String	50
17	User Choice 7	userChoice7	Optional	String	50
18	User Choice 8	userChoice8	Optional	String	50
19	User Choice 9	userChoice9	Optional	String	50
20	User Choice 10	userChoice10	Optional	String	50
21	User Choice 11	userChoice11	Optional	String	250
22	User Choice 12	userChoice12	Optional	String	250
23	User Choice 13	userChoice13	Optional	String	250

24	User Choice 14	userChoice14	Optional	String	250
25	User Choice 15	userChoice15	Optional	String	250
26	User Choice 16	userChoice16	Optional	String	250
27	User Choice 17	userChoice17	Optional	String	250
28	User Choice 18	userChoice18	Optional	String	250
29	User Choice 19	userChoice19	Optional	String	250
30	User Choice 20	userChoice20	Optional	String	250
31	User Choice 21	userChoice21	Optional	String	250
32	User Choice 22	userChoice22	Optional	String	250
33	User Choice 23	userChoice23	Optional	String	250
34	User Choice 24	userChoice24	Optional	String	250
35	User Choice 25	userChoice25	Optional	String	250
36	Payment Method	paymentMethod	Optional	String	6
37	Street One	streetOne	Optional	String	50
38	Street Two	streetTwo	Optional	String	50
39	City	city	Optional	String	20
40	State	state	Optional	String	2
41	Zip	zip	Optional	String	10
42	Country	country	Optional	String	20
43	Day time Phone	daytimePhone	Optional	String	20
44	Night time Phone	eveningPhone	Optional	String	20
45	Email	email	Optional	String	50
46	Redirect URL	redirectUrl	Optional	String	100
47	Redirect URL Parameters	redirectUrlParameters	Optional	String	NA
48	Retries Allowed	retriesAllowed	Optional	String	1
49	Content Embedded (3)	contentEmbedded	Optional	"true"	4

50	Time Stamp	timestamp	Yes	String	13
51	Key	NA	No	NA	NA
NA	Hash	hash	Yes	String	NA

Parameter Descriptions

Order Type – The Order Type you pass into Commerce Manager must correspond to one you configured using the Commerce Manager Administrator - Add New Order. This relationship will ensure that the order’s payment options – including required fields and processors – selected when creating the new order, are applied when a payer is passed in with this Order Type.

Order Number – The account being paid or the unique identifier of the order. This can be displayed to the payer during their session or simply captured and sent back with the transaction.

Order Name – The name of the order or the name of the payer.

Order Description – This is the description of the order. The Order Description displays in various locations in Commerce Manager.

Amount – The amount that the payer should be paying. This will populate the Payment Amount field in Commerce Manager.

Order Fee – The Order Fee can be set using the Edit Order form or is can be passed into Commerce Manager during transaction processing. If an Order Fee is passed into Commerce Manager it will overwrite an Order Fee rule established using the Edit Order form.

Current Amount Due – The Current Amount Due can be a different monetary value passed in during the Transaction Processing representing a different calculation than the Amount. The Current Amount Due will not be used for the Payment Amount but can be displayed for informational purposes or can be applied to another field and used for additional order payment rules.

Balance – The Balance can be a different monetary value passed in during the Transaction Processing representing a different calculation than the Amount. The Balance will not be used for the Payment Amount but can be displayed for informational purposes or can be applied to another field and used for additional order payment rules.

Current Balance – The Current Balance can be a different monetary value passed in during the Transaction Processing representing a different calculation than the Amount. The Current Balance will not be used for the Payment Amount but can be displayed for

informational purposes or can be applied to another field and used for additional order payment rules.

Due Date – The due date for the order can be displayed if passed into Commerce Manager. No logic can be performed on this parameter; it is for display purposes only.

User Choice 1-20 – Twenty discretionary fields that can be either passed into Commerce Manager or captured during the Transaction Processing, depending on how the order is configured.

Payment Method – See Appendix F for information on implementing the Payment Method feature.

Payment Form Information – Payment form information collected before the payer has been sent to Commerce Manager can be passed in to pre-populated payment forms. This includes:

- streetOne
- streetTwo
- city
- state
- zip
- country
- daytimePhone
- eveningPhone
- email

Redirect URL – See [Appendix D: Optional Feature – Redirect URL](#) for instructions on implementing the redirectUrl feature. The redirectUrl parameters are:

- redirectUrl
- redirectUrlParameters
- retriesAllowed

Content Embedded – See [Appendix E: Optional Feature – Embedding Commerce Manager in an iframe](#) for instructions on implementing the Content Embedded Commerce Manager feature.

Time Stamp – The Time Stamp is a long integer number of the millisecond from the year 1970 (epoch time). See [Appendix C: Generating the Time Stamp Parameter](#) for additional information about generating the Time Stamp.

Key – The Key value that is used along with the Order Type and the Time Stamp to generate the hash digest which will be used to verify the authenticity of the payer being passed into Commerce Manager. Longer, randomly generated keys should be used to maximize security. Key administration options in Commerce Manager include an option

to generate a random key. When a new order is migrated to the production Commerce Manager environment, a new production key will need to be generated.

Hash – The Hash is a product of the concatenation of the Order Type, Time Stamp and Key being passed through your hash (SHA-256) library/function; if optional parameters are being passed into Commerce Manager, they will need to be hashed as well.

Appendix B: Secure SHA-256 Hash

What is an SHA-256 hash?

SHA-256 hash is a one-way function that takes bytes as input and output bytes that represent a “fingerprint” or “signature” of the input.

What does this algorithm accomplish for our transaction processing?

It ensures that only your school can send Commerce Manager requests to process a transaction. A user could not type the URL into the browser and access Commerce Manager.

How does a hash function work?

The input of the SHA-256 hash function will be the concatenated string of parameter values you are passing to Commerce Manager plus the time stamp and a “key”. The output of the function will be the hash value (digest) that you will pass to Commerce Manager as an additional parameter.

- The source site and destination site will agree on a secret phrase.
- The source site assembles the raw data that needs to be passed.
- The source site appends the secret phrase to the raw data and produces the digest using the SHA-256 hashing algorithm.
- The source site produces the URL with the original data and the digest, but without the secret “key”.
- The source site appends the query string and redirects the URL to the destination site.
- When the destination site receives the URL, it extracts the raw data and digest.
- The destination site appends the secret phrase to the raw data and produces the digest using the SHA-256 hashing algorithm.
- The resulted digest is compared with the digest in the URL. If they are the same, then the URL is authenticated.

- The destination site processes the raw data.

Appendix C: Generating the Time Stamp Parameter

The timestamp value is the difference, measured in milliseconds, between the current time and midnight, January 1, 1970 UTC (or GMT). This is known as epoch time.

Below is an example of generating the timestamp in the Java environment:

```
//obtain the epoch time in milliseconds  
  
long timestamp = System.currentTimeMillis();
```

It is important to verify that the source system time is close to the official time. A good reference for official time and other time related information is the Time Service Department of the U.S. Naval Observatory website. Its URL is:

<http://tycho.usno.navy.mil/>

Appendix D: Optional Feature – Redirect URL

Using the Redirect URL feature, after the payment is processed successfully, Commerce Manager will redirect the user to the institution's web site along with a list of requested parameters. This provides the institution with the transaction's final status (success, decline, etc.) and information needed to generate a receipt; no receipt is displayed in Commerce Manager. In addition to the order parameters three additional parameters must be hashed and sent to Commerce Manager:

- `redirectUrl`
- `redirectUrlParameters`
- `retiresAllowed`

Example of redirect portion in pass-through url:

```
redirectUrl=https://test.testname.com/crm/PaymentGatewayServlet/QuikPAY/callback&redirectUrlParameters=transactionType,transactionStatus,transactionId,transactionTotalAmount,transactionDate,orderNumber,orderType,payerType,payerIdentifier,payerFullName,email
```

redirectUrl – URL to which Commerce Manager payer will be redirected to view the receipt.

redirectUrlParameters – Please select which attributes you wish to receive for shopping cart redirect to your system. Please note that all attribute names are case sensitive. Parameters are requested by sending a comma delimited list of transaction attribute names that you want to receive. Use value "none" if you do not want to receive

any attribute. The attributes you request are returned as name value pairs in a random order.

retriesAllowed – This parameter sets the number of retries you will allow the payer to attempt if the first payment attempt was not successful.

Attribute Name	Description
transactionType	1 - Credit Card Payment 2 - Credit Card Refund 3 - eCheck Payment
transactionStatus	If Transaction Type field value is 1 or 2 (credit card payment/refund): 1 - Accepted credit card payment/refund (successful) 2 - Rejected credit card payment/refund (declined) 3 - Error credit card payment/refund (error) 4 - Unknown credit card payment/refund (unknown) If Transaction Type field value is 3 (eCheck payment): 5 - Accepted eCheck payment (successful) 6 - Posted eCheck payment (successful) 7 - Returned eCheck payment (failed) 8 - NOC eCheck payment (successful)
transactionId	Unique identifier generated by QuikPAY® at the time of the transaction, often referred to as Confirmation Number.
originalTransactionId	The original payment transaction id if the transaction is a credit card refund.
transactionTotalAmount	Actual transaction total amount, the sum of values of Order Amount and Order Fee fields.
transactionDate	Date when the transaction was captured by QuikPAY® in YYYYMMDDHHMM format.
transactionAccountType	The type of account used to make the payment: e.g. "VISA", "DISCOVER", "CHECKING", "SAVINGS"
transactionEffectiveDate	Business day that the transaction belongs to in YYYYMMDDHHMM format. The hours and minutes, HHMM, will always be 0000 for the Transaction Effective Date.
transactionDescription	Description of the transaction.
transactionResultDate	Date that the transaction was processed in YYYYMMDDHHMM format. If Transaction Status field value is 7 or 8, this will be the date that QuikPAY® was notified of the result by the NACHA network.
transactionResultEffectiveDate	Business day that the result of the transaction would be effective in YYYYMMDDHHMM format. The hours and minutes, HHMM, will always be 0000 for the Transaction Result Effective Date. If Transaction Status field value is 7 or 8, this will be the date that NACHA network generated the result.

transactionResultCode	if Transaction Status field value is 1 - credit card authorization code if Transaction Status field value is 2 - credit card rejection code if Transaction Status field value is 3 or 4 - credit card processing error code if Transaction Status field value is 5 or 6 - blank if Transaction Status field value is 7 - NACHA return code if Transaction Status field value is 8 - NACHA change code
transactionResultMessage	Detailed message describing the result of the transaction
orderNumber	The account being paid or the unique identifier of the order
orderType	Account Type being paid
orderName	Name of the Order
orderDescription	Description of the order
orderAmount	The amount the user wished to pay originally
orderFee	Add-on fee if any, often referred to as convenience fee
orderAmountDue	The amount due on the order
orderDueDate	Due date of the order in YYYYMMDDHHMM format
orderBalance	The balance on the order
orderCurrentStatusBalance	The balance that is obtained from Current Account Status (if the client has this feature implented)
orderCurrentStatusAmountDue	The amount due that is obtained from Current Account Status (if the client has this feature implented)
payerType	Type of the payer. Example: "university_campus_payer".
payerIdentifier	The unique identifier for the user within the institution. This is typically a student ID.
payerFullName	Full name of the payer. Example: "John Smith"
actualPayerType	Type of the actual payer. It may be the same as PayerType field. Other example: "university_campus_authorized".
actualPayerIdentifier	The unique identifier for the actual payer.
actualPayerFullName	Full name of the actual payer.
accountHolderName	The name on the checking or savings account for eCheck transaction or the name on the credit card for credit card.
creditCardLastFour	Last four digits of the credit card used for the transaction.
streetOne	Street line 1
streetTwo	Street line 2
city	City
state	State

zip	Zip code
country	Country
daytimePhone	Daytime phone number
eveningPhone	Evening phone number
email	email address that the user entered on the payment form
userChoice1	Optional field 1.
userChoice2	Optional field 2.
userChoice3	Optional field 3.
userChoice4	Optional field 4.
userChoice5	Optional field 5.
userChoice6	Optional field 6.
userChoice7	Optional field 7.
userChoice8	Optional field 8.
userChoice9	Optional field 9.
userChoice10	Optional field 10.
userChoice11	Optional field 11.
userChoice12	Optional field 12.
userChoice13	Optional field 13.
userChoice14	Optional field 14.
userChoice15	Optional field 15.
userChoice16	Optional field 16.
userChoice17	Optional field 17.
userChoice18	Optional field 18.
userChoice19	Optional field 19.
userChoice20	Optional field 20.
userChoice21	Optional field 21.
userChoice22	Optional field 22.
userChoice23	Optional field 23.
userChoice24	Optional field 24.
userChoice25	Optional field 25.

 Parameter rules:

Payer Parameters – payerType, payerIdentifier and payerFullName must all be requested together. If you request one, you must request all three.

Actual Payer Parameters – actualPayerType, actualPayerIdentifier and actualPayerFullName must all be requested together. If you request one, you must request all three

Appendix E: Optional Feature – Embedding Commerce Manager in an iframe

Including the optional Content Embedded parameter with a “true” value in transaction processing will result in the transaction session being displayed without the Commerce Manager banner logo or the Main Menu being displayed. Thus, Commerce Manager can be embedded in another web page or application.

One way of embedding a web document inside another page or application is using an

The screenshot shows a web page titled "Your Shopping Cart" with a "Checkout" section. A table lists items: Red T-Shirt (1, \$15.00) and Blue T-Shirt (2, \$15.00). Below the table, the total is \$30.00, plus tax of \$5.00, for a final total of \$50.00. A red-bordered box highlights a "Enter Payment Amount" form. The form contains the text "Please enter in the amount you want to pay and click 'Continue' button." and a sub-form with "Account: Shopping Cart Order", "Payment Amount: \$50.00", and "Payment Method: Select one..." dropdown. A "Continue" button is at the bottom right of the form. A red arrow points to the bottom left of the form box, with the text "QUIKPAY(R) EMBEDDED WITHIN AN IFRAME." below it.

Item Description	Qty	Each	Total
Red T-Shirt	1	\$15.00	\$15.00
Blue T-Shirt	2	\$15.00	\$30.00
			Tax: \$5.00
			Total: \$50.00

Enter Payment Amount
Please enter in the amount you want to pay and click "Continue" button.

Account: Shopping Cart Order
Payment Amount: \$50.00
Payment Method: Select one...
Continue

QUIKPAY(R) EMBEDDED WITHIN AN IFRAME.

iframe element to create an inline frame containing Commerce Manager.

Here are some considerations for embedding Commerce Manager:

- The width of an embedded page is 605 pixels.
- The heights of embedded pages change based on which Commerce Manager pages are being displayed during the payment flow. A page's height will vary based on how much information is being collected on a page. If a decision is made to limit the size of the frame then scrolling should be turned on.

Sample Code – embedding Commerce Manager using an iframe element

```
<iframe id="shoppingCartFrame" width="620" height="700"
src="<%=request.getContextPath()%>https://quikpayasp.com/quikpay_uiversity/comme
rce_manager/payer.do?orderType=shopping_cart&contentEmbedded=true"
frameborder="yes" scrolling="no">

</iframe>
```

Appendix F: Optional Feature – Payment Method Flag

The Payment Method flag can be passed into Commerce Manager to control which payment methods are shown to the payer during their session. If a Commerce Manager Order has been created that has been linked to both a credit card and an eCheck processor, the Payment Method can be used for additional control.

Scenario 1 – if the Payment Method flag is set to eCheck only (paymentMethod=ach), a payer can be limited to the eCheck payment option during that specific session.

Scenario 2 – if the Payment Method flag is set to Credit Card only (paymentMethod=cc), a payer can be limited to the Credit Card payment option during that specific session.

Scenario 3 – if the Payment Method flag is set to both eCheck and Credit Card (paymentMethod=ach_cc), a payer will be given both payment options during that specific session.