TRANSACT

eMarket

Payment Integration Guide

Legacy Integration Only

Applicable for approvals requested prior to 2024.

October 2024

Trademarks & Copyright

Copyright © 2024. Transact Campus Payments, Inc. All rights reserved.

Disclaimers

This document contains confidential information and is intended for distribution to authorized external parties only.

The content of this document is provided for informational use only and is subject to change without notice. Transact assumes no liability or responsibility for any errors or inaccuracies that may appear in this document.

Contact Information

For information and support prior to the deployment process, contact your Payments Project Team.

For post-deployment support, contact Payments Support at 1-888-381-8054 or log in to Transact Connect.

About This Document

This document explains the process for integrating your campuses or third-party vendor's payment application with Transact Payments to process payments.

The information included in this document is intended for integrators already using eMarket Checkout and APIs.

Scope

Examine and highlight the differences between eMarket's two external integration options, Checkout and Gateway, while outlining the specific requirements for integrating each option.

Audience

This document is directed at clients interested in:

- Assessing the integration requirements for eMarket Checkout and Gateway.
- Reviewing the differences between eMarket Checkout and Gateway.
- Performing integration steps for Transact Payments to process payments as a Checkout or Gateway.
- Training their staff in the integration and ongoing maintenance requirements for their eMarket Checkout or Gateway.

Glossary

The table below defines terms relevant to this document's content.

Table 1: Document Glossary

TERM	DEFINITION
eMarket	Transact Payments eCommerce module for selling items for departments and other campus entities through online marketplaces.
Checkout	eMarket option in which customers are transferred to the Transact Payments system for payment collection and processing but remain in the campuses or vendor's application for item selection.
Gateway	eMarket option in which customers' payment information is transferred to the Transact Payments system for payment processing but remain in the user interface of the campus or vendor's application for the entire checkout process.
Storefront	eMarket option in which customers are transferred to the Transact Payments system for item selection, payment collection and processing.
Transact Payments Administration	Refers the Transact Payments Administration site, through which Transact Payments modules, including eMarket, are configured.

References

This document refers to the following documents and other resources.

Table 2: Referenced Documents

TITLE	AUTHOR	URL / FILE LOCATION
eMarket Store Setup Manual	Transact	Support Library > Modules and Services > ePayment and eMarket
ePayment & eMarket User Manual	Transact	Support Library > Modules and Services > ePayment and eMarket
System Setup User Manual	Transact	Support Library > Modules and Services > System Administration
Transaction Inquiry Service Quick Guide	Transact	Support Library > Modules and Services > Inquiry and Reporting
HTML URL Encoding Reference	w3schools.com	http://www.w3schools.com/tags/ref_urlencode.asp

Contents

1 Administrative Requirements 6
1.1 Mutual Non-Disclosure Agreement 6
1.2 Module License 6
1.3 Integration Development 6
1.4 Integration Testing 6
2 Checkout & Gateway Overview 8
2.1 Comparison of Checkout & Gateway 8
2.2 Payment Processing Overview 9
2.3 Checkout Overview 10
2.4 Gateway Overview 11
3 Checkout Integration Requirements 13
3.1 Checkout Payment Requests to Transact Payments 13
3.2 Checkout Payment Notifications from Transact Payments 15
4 Additional Checkout Details 18
4.1 Checkout Request Security 18
4.2 Redirection via Sign-Out URL 20
4.3 Receipt Display Option 20
4.4 Checkout User Interface 20
4.5 API for Transaction Inquiries 20
5 Reference Types 21
5.1 Transaction-Level Reference Types 21
5.2 Item-Level Reference Types 23
6 Gateway Integration Requirements 25
6.1 Gateway Payment Requests to Transact Payments 25
6.2 Gateway Payment Responses from Transact Payments 28
6.3 Gateway Transaction Refunds 29
Appendix A Checkout Requests 31
Appendix B Checkout HTTP Notifications 35
Appendix C Gateway Requests 42
Appendix D Gateway Responses 46

1 Administrative Requirements

Prior to integrating your application with Transact Payments eMarket as a Checkout or Gateway, you must fulfill the following requirements.

1.1 Mutual Non-Disclosure Agreement

Before sharing proprietary information related to the integration, Transact requires the campus or third-party vendor to sign and submit a Mutual Non-Disclosure Agreement (NDA).

In most cases, Transact will already have a signed NDA on file for your organization. However, **if you are a third-party vendor that has NOT already integrated with Transact Payments, you will need to submit a signed NDA to Transact prior to the integration.** A Transact representative will provide you with the NDA form.

Note: We mandate that your campus's staff NOT share this *eMarket Checkout & Gateway Payment Integration Guide*, or any other Transact documentation, with third-party vendors, without first receiving approval from a Transact representative.

1.2 Module License

To configure an eMarket Checkout or Gateway, your campus must possess either a license for unlimited eMarket or an individual eMarket license for each Checkout or Gateway integration.

Note: eMarket also offers a *Storefront* option in which customers select and purchase items entirely through the Transact Payments site. For details, contact a Transact Payments representative.

1.3 Integration Development

Your integration with Transact Payments is based on an open HTTP standard, and you can develop your integration in any language of your choice. Most modern programming languages provide the tools required to call the Transact Payments API and process the information sent back from Transact Payments.

Note: Transact Payments may redirect your HTTP call multiple times before rendering the service. When making HTTP calls, ensure that automatic forwarding is enabled.

1.4 Integration Testing

You will perform all the testing for your integration in the campus's Transact Payments test instance.

This Transact Payments test instance is a sandboxed environment in which you can securely test the payment process through the campus environment. The sandbox is connected to a mock credit card terminal through Transact Payments. Using the test instance, you can perform multiple testing scenarios, including various credit card errors. The Transact Consulting Team can provide you with test credit card numbers.

In certain cases, third-party vendors also request to test basic integration with the Transact Payments application prior to testing through the sandbox in the campus environment. Transact Payments can accommodate this preference by providing a separate sandbox in the Transact Payments environment. Contact a Transact representative for details.

Ensure you determine your eMarket type—either Checkout or Gateway—before you begin testing.

2 Checkout & Gateway Overview

Transact Payments offers two distinct solutions to enable your campuses or third-party vendor's web application to accept and process payments. These solutions are known as *Checkout* and *Gateway*, and any eMarket license may be configured as one of these options.

2.1 Comparison of Checkout & Gateway

The table below highlights the primary differences between Checkout and Gateway.

Table 3: Checkout and Gateway Comparison

INTEGRATION REQUIREMENT	CHECKOUT	GATEWAY
What level of effort is required of the campus or vendor to comply with PCI DSS?	**Low.** The campus's and/or vendor's PCI scope is significantly reduced, as all sensitive information, such as credit card numbers, is entered in and transmitted through the Transact Payments Checkout.	**High.** The campus and/or vendor is responsible for complying with PCI DSS for collecting sensitive payment information such as credit card numbers.
Which website does the customer use for item selection?	**Campus or vendor website.**	**Campus or vendor website.**
Which website does the customer use for payment data entry?	**Transact Payments website.**	**Campus or vendor website.**
What level of control does the campus or vendor have over the application's look and feel and payment flow?	**Moderate.** Provides multiple payment flow options and some customization to match the application's look and feel. However, the UI may not be completely identical.	**High.** Provides full control over the look and feel and payment flow as the customer never leaves the campus or vendor's website.
What level of technical skills must the campus or vendor have to implement the payment integration?	**Moderate.** Must be familiar with HTML and forms.	**High.** Requires programming skills in a language such as Perl, Java, or VB.
How much time is required to implement the payment integration?	**Moderate.** Generally, less than a Gateway.	**High.** Generally, more than a Checkout.
Can I use SmartPay with the integration?	**Yes.** SmartPay is supported for credit cards.	**No.** SmartPay is not supported.

2.1.1 Common Situations for Checkout vs. Gateway

The following table presents examples of situations where a Checkout or a Gateway would best fit a campus's needs.

Table 4: Common Scenarios for Checkout and Gateway Integration

CHECKOUT	GATEWAY
✓ Your campus has an existing website that accepts payment information, which is processed manually or via a credit card system other than cashnet.com. However, your vendor or department no longer has the mandate or desire to handle any part of the credit card process. ✓ Your campus does not have a website yet but wants to keep control of the information flow and look and feel of the site up to the point that the customer begins the credit card payment process.	✓ Your campus has an existing website that accepts payment information, which is processed manually or via a credit card system other than cashnet.com. ✓ Your campus is using a PCI-compliant third-party application and would like to direct all credit card payments through one centralized system.

2.2 Payment Processing Overview

Like most applications that process payments, the payment flow for both eMarket Checkout and Gateway consists of three main phases:

1. **Item Selection:** In both Checkout and Gateway, your campus website or vendor is responsible for providing the functionality customers will use to select items for purchase. You may implement this functionality in any manner you deem appropriate.

   Examples range from a simple prompt for a donation to complex diagrams displaying the tickets with available seats in a football stadium. The user interface may require login authentication, or it may be open to the general public. All these decisions are up to you or your third-party vendor.

2. **Payment Collection & Processing:** The process for collecting payment information will vary depending on whether you are using a Checkout or Gateway, as described in the Checkout Overview (p. 10) and Gateway Overview (p. 11) sections.

   In both cases, however, the shopping cart information must be sent to Transact Payments in TLS-encrypted HTTPS messages and Transact Payments will process the payment.

3. **Post-Payment Processing:** Following payment processing, Transact Payments will notify your application of the successful payment. Additionally, in a Checkout, the customer may be presented a receipt or transferred back to your application, depending on how you choose to configure the Checkout.

2.3 Checkout Overview

When integrating your application with Transact Payments eMarket as a Checkout, you hand over the responsibility for collecting sensitive payment information, such as the credit card and bank account numbers, to the Transact Payments hosted environment, thereby reducing your scope for PCI DSS compliance. You also maintain a moderate amount of control over your application's workflow and the look and feel of the user interface.

The following diagram outlines the payment flow for eMarket Checkout. As shown below, the customer is transferred to the Checkout after selecting items. The customer then enters payment information in the Checkout and, after the payment is processed, may be transferred back to the third-party application.

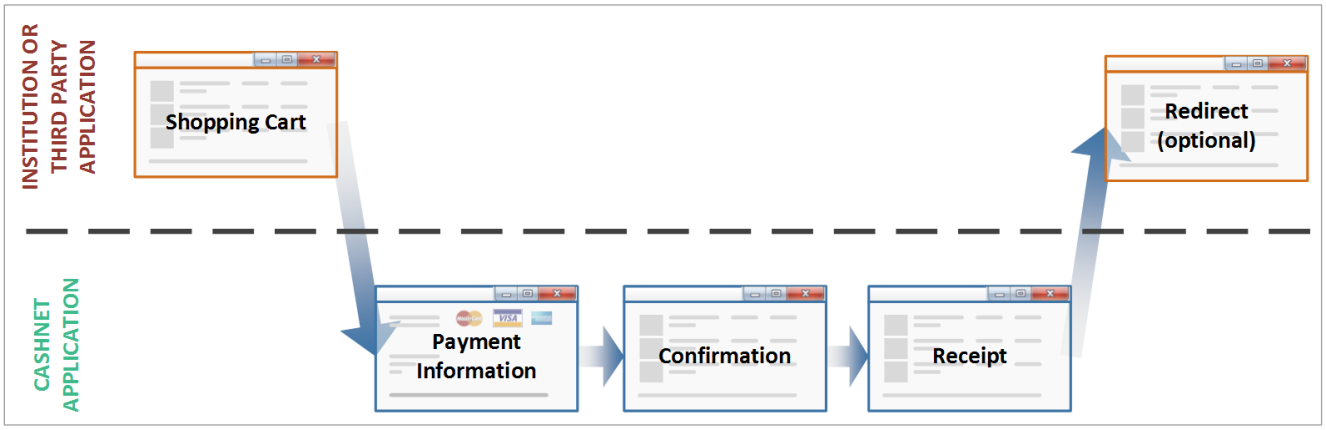

Figure 1 Checkout Payment Processing Flow

2.3.1 Checkout Webpage Sample

The following code provides an example of a form populated through a Checkout web page, which you would use to create your Checkout request.

In the web page, the customer has accessed the Psychology Department website and selected to purchase a ticket for a conference ($124.00) and a lunch at the conference ($15.60).

```
<html>
  <head>
    <title>Our Checkout</title>
  </head>
  <body>
    <form method="get"
action="https://commerce.cashnet.com/CheckoutName">
      <input type="hidden" name="itemcode" value="PSYCH-CONF"><br>
      <input type="hidden" name="amount" value="124.00"><br>
      <input type="hidden" name="itemcode2" value="PSYCH-LUNCH"><br>
      <input type="hidden" name="amount2" value="15.60"><br>
      <input type="submit" name="submit" value="Make Payment">
    </form>
  </body>
</html>
```

When the customer clicks **Make a Payment**, he will be redirected to the cashnet.com website to enter payment information and confirm the transaction.

Once the payment has been processed, you will receive one or multiple payment notifications, depending on your preference, which are detailed in Checkout Payment Notifications from Transact Payments on page 15.

2.4 Gateway Overview

When integrating your application with Transact Payments eMarket as a Gateway, you have full control over the look and feel of the website. The Gateway itself is only a web-based HTTPS API that submits and accepts required parameters for payment processing, including the credit card information, billing address, amount to be charged, and other information. Upon obtaining the payment information, Transact Payments authorizes and settles the payment.

Important Note:

If you are using a Gateway, your application must collect the sensitive payment information from the customer, including credit card and bank account numbers. Although the Transact Payments hosted platform is PCI DSS compliant and will securely authorize and process the payment, the application and merchant are responsible for complying with the appropriate security standards when collecting payment information.

The following diagram outlines the payment flow for an eMarket Gateway. As shown below, the customer remains in the third-party website throughout the application flow, while the Gateway API call occurs in the background to perform the credit card and other payment processing.

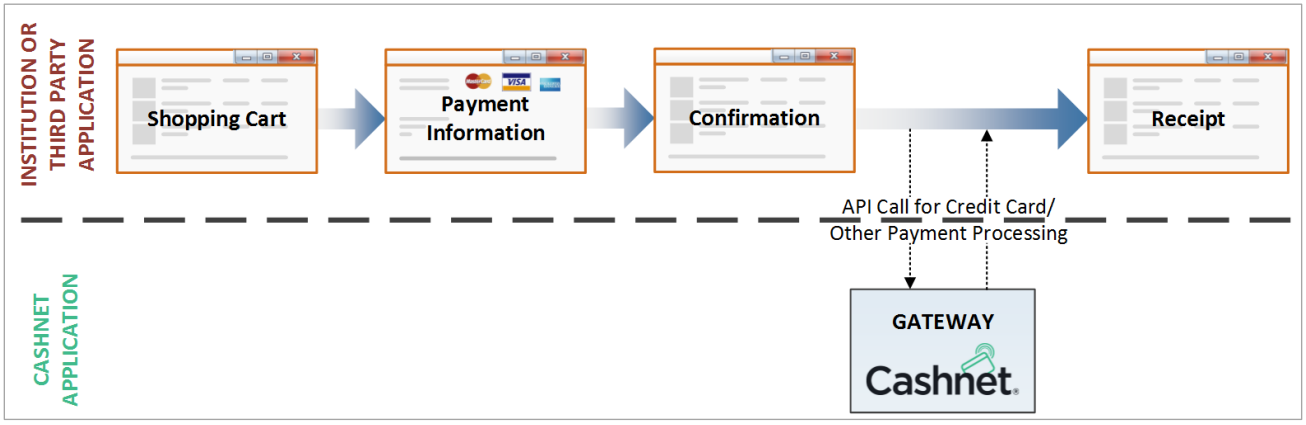

Figure 2 Gateway Payment Processing Flow

2.4.1 Gateway Webpage Sample

The following code provides an example of a form populated through a Gateway web page, which you would use to create your Gateway request.

In the web page, the customer has accessed the Psychology Department website and selected to purchase a ticket for a conference ($124.00) and a lunch at the conference ($15.60). The customer has entered a credit card number to make the payment.

```
<html>
  <head>
    <title>Our Gateway</title>
  </head>
  <body>
    <form method="post" action="https://commerce.cashnet.com/GatewayName">
      <input type="hidden" name="merchant" value="GATEWAY"><br>
      <input type="hidden" name="command" value="SALE"><br>
      <input type="hidden" name="custcode" value="987654321"><br>
      <input type="hidden" name="operator" value="GWUSER"><br>
      <input type="hidden" name="password" value="secret"><br>
      <input type="hidden" name="station" value="GW1"><br>
      <input type="hidden" name="itemcode" value="PSYCH-CONF"><br>
      <input type="hidden" name="amount" value="124.00"><br>
      <input type="hidden" name="itemcode2" value="PSYCH-LUNCH"><br>
      <input type="hidden" name="amount2" value="15.60"><br>
      <input type="hidden" name="cardno" value="Card Number"><br>
      <input type="hidden" name="expdate" value="1215"><br>
      <input type="hidden" name="submit" value="Submit">
    </form>
  </body>
</html>
```

When the customer clicks **Submit**, his transaction and payment information will be sent to the Transact Payments Gateway for payment processing.

Once the payment has been processed, you will receive a synchronous response. Responses are detailed in Gateway Payment Responses from Transact Payments on page 28. Below is an example successful response.

```
<cngateway>
  result=0&
  tx=1234&
  busdate=6/25/2015&
</cngateway>
```

3 Checkout Integration Requirements

There are two main requirements for your application to communicate with the eMarket Checkout: (1) you must send the Transact Payments application payment requests containing the required transaction information; and (2) you must elect a method to receive payment notifications from the Transact Payments application.

Note: This section only applies to eMarket Checkout integrations. For eMarket Gateway integration requirements, see Gateway Integration Requirements on page 25.

3.1 Checkout Payment Requests to Transact Payments

When a customer selects items or places items in the shopping cart, your application makes the first call to the Transact Payments application. The customer is then transferred to Transact Payments Checkout along with the input parameters required to perform payment processing.

3.1.1 Checkout Request Syntax

Your Checkout will be assigned a unique URL to communicate with Transact Payments. You will send Transact Payments transaction information in this URL in standard input parameters, formatted in query string name-value pairs separated by an ampersand (&).

The query string can be sent through HTTPS POST or GET. The Transact Payments application can parse your form input data. Depending on whether you choose to send it as HTTPS GET or POST, the URL will be slightly different.

HTTPS GET String:

```
https://commerce.cashnet.com/CheckoutName?
  name=value&
  name=value
```

Note: In GET format, you must convert symbols into acceptable ASCII characters; accordingly, all GET examples in this guide will be properly encoded.

HTTPS POST Format:

In POST format, you will send your Checkout name in the `virtual` parameter.

```
https://commerce.cashnet.com/404Handler/pageredirpost.aspx?
  virtual=CheckoutName&
  name=value&
  name=value
```

Note: In POST format, you do NOT need to convert symbols into ASCII characters; accordingly, all POST examples in this guide will NOT be encoded.

3.1.2 Required Checkout Request Information

For Transact Payments to determine the payment details, your requests must contain the **shopping cart details**, or information about the items selected to purchase. Shopping cart details must include the Item Codes, and may include the amounts and other information, such as quantity or G/L Code.

For the complete list of parameters that can be sent in the Checkout request, as well as multiple request examples, see Appendix A Checkout Requests on page 31.

Below is an example request containing three items and their associated amounts.

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=APPFEE&
    amount=50.00&
  itemcode2=PARKING&
    amount2=150.00&
  itemcode3=TRANSCRIPT&
    amount3=50.00
```

As the example shows, when the request has multiple Item Codes, the first parameter is `itemcode`, the second increments to `itemcode2`; the third, `itemcode3`; and so forth. Furthermore, the associated `amount` parameters or other item-level must all have the appended number that matches the associated Item Code.

3.1.3 Additional Checkout Information

In your requests to Transact Payments, you may also send additional information to use for reporting and identifying transactions. Examples include the session ID and tax amount.

This information can be sent in custom parameters, which are known in Transact Payments as *Reference Types* and can be associated with either the entire transaction or specific items. For detailed information, see the Reference Types section on page 21.

Below is an example GET request for one item containing one transaction-level Reference Type:

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=APPFEE&
    amount=50.00&
  SESSIONID=2234ffa
```

3.1.4 Security for Checkout Requests

Because eMarket Checkout does not restrict the source of Checkout requests, request tampering is a potential security concern, which could occur if a user willfully modified a Checkout request to contain a different payment amount in an effort to pay less than the amount due.

Note: Checkout does not restrict the source of requests so that you can have multiple departments using the same eMarket Checkout site.

To prevent the possibility of request tampering, Transact Payments enables you to set Item Code Default Amounts (p. 18) or use Hashing & Message Digests (p. 18). For details, see Checkout Request Security (p. 18).

3.2 Checkout Payment Notifications from Transact Payments

Transact Payments provides five (5) methods for notifying your application of payments, most are asynchronous calls.

3.2.1 Available Notification Types

You may select any notification option you prefer—or combination of notification options—and communicate it to your Transact Consulting Team, who can also provide additional details if you have further questions.

Important Notes:

- ✓ Notification types marked with an asterisk (*) are recommended because they have guaranteed delivery. For details, see the Comparison of Notification Types below.
- ✓ Notification types marked with an asterisk (*) may incur additional charges. For details, contact your Transact Consulting Team or Client Manager.
- ✓ Third-party vendors—except for T2—usually use HTTP Notifications.

1. **HTTP Notifications:** Transact Payments sends a synchronous HTTP request (fire-and-forget) to a URL of your choice. You can configure separate URLs for successful and unsuccessful transactions. For the list of parameters that may be sent in HTTP Notifications, refer to Appendix B on page 35.
2. * **Online Extract (aka Post):** Transact Payments performs an asynchronous call to your application once credit card authorization is completed, and the customer is immediately transferred to the receipt page. You will receive the notification by the time the customer is transferred back to your application.
3. * **Synchronized Online Post**: Transact Payments performs an asynchronous Online Extract call, as explained above; however, the customer will remain in the "Payment processing" screen until the ERP or originating system has returned a success message. This is most commonly used when the transaction is not the last step (e.g., the customer selects seats after paying for the concert).
4. **Sign-Out URL:** After making a payment in Transact Payments, customers are redirected to a URL. The sign-out URL is constructed according to your specifications, containing the payment parameters you want to receive. You may configure the URL as a static field or send it as a parameter (`signouturl`) when transferring the customer to the Transact Payments application. Transact Payments posts information back to the URL in standard query format as HTTPS GET or POST. You can configure separate URLs for successful and unsuccessful transactions. For the list of parameters that may be sent in Sign-Out URL, refer to Appendix B on page 35.
5. * **Batch File:** You will receive payment notifications in a nightly file for transactions occurring in that business day. The batch file can be downloaded from Transact's secure FTP server. Transact Payments supports both fixed width and delimited batch file formats. Most schools use batch files in conjunction with notification types #2 and #3 as a way to perform independent verification for the day's activity.

3.2.2 Comparison of Notification Types

The following table explains the advantages and disadvantages of each notification type.

Table 5: Advantages and Disadvantages of Payment Notification Types

	TYPE	ADVANTAGES	DISADVANTAGES
1	HTTP Notifications	✓ Your application will be notified before the customer is transferred out of Transact Payments or receives a receipt. ✓ Logging is supported in Transact Payments.	× If Transact Payments cannot reach the URL, we will NOT retry the request. However, the custom will still receive a receipt.
2	Online Extract	✓ Guaranteed delivery. Transact Payments will try to call your system up to 4 times. You may also re-attempt to send extracts in Transact Payments at any time. ✓ Transact Payments provides tools to monitor the status of online extract notifications. ✓ Parameters and their names are not fixed and can be configured through Transact Payments extract tools. ✓ Nearly all the data fields captured in transactions and stored in Transact Payments can be extracted to your system. ✓ Transact Payments supports both HTTPS messaging and web services as communication methods. ✓ Logging is supported in Transact Payments.	× Because the notification is kept in a queue and processed sequentially, there is no guarantee that notification will be sent before the customer is transferred back to your application. × Payment notifications will only be sent for successful transactions; notifications will NOT be sent for unsuccessful transactions such as an invalid credit card number.

	TYPE	ADVANTAGES	DISADVANTAGES
3	Synchronized Online Post	✓ Your application will be notified before the customer is transferred out of Transact Payments or receives a receipt. As such, this is ideal if your system has a business process that requires you to receive a payment notification before redirecting the customer. ✓ Transact Payments provides tools to monitor the status of online extract notifications. ✓ Parameters and their names are not fixed and can be configured through Transact Payments extract tools. ✓ Nearly all the data fields captured in transactions and stored in Transact Payments can be extracted to your system. ✓ Transact Payments supports both HTTPS messaging and web services as communication methods. ✓ Logging is supported in Transact Payments.	× Payment notifications will only be sent for successful transactions; notifications will NOT be sent for unsuccessful transactions such as an invalid credit card number. × If your server is down during the transaction, then the user will be stuck in Transact Payments.
4	Sign-Out URL	**Although Sign-Out URL is available, we do NOT recommend using it. Refer to the disadvantages to the right.**	× If the customer closes the browser or navigates away directly after making a payment, you may not receive a payment notification, but the customer's credit card will be charged, and the transaction recorded in Transact Payments. × Logging is NOT supported in Transact Payments. × Potential security risk when passing data.
5	Batch File	✓ Guaranteed delivery. ✓ Logging is supported in Transact Payments. ✓ You may also re-attempt to send batch files through Transact Payments at any time.	× The information is not sent in real-time. × You may only receive one file per business day.

4 Additional Checkout Details

For your Checkout, you may use the following additional options.

4.1 Checkout Request Security

To secure your Transact Payments Checkout requests, Transact provides Item Code default amounts or hashing and message digests.

4.1.1 Item Code Default Amounts

In the Transact Payments Administration site, you can set a default amount in Item Code configuration, ensuring that the amount charged is determined from the value set in the Transact Payments database and not from the Checkout request. Item Code default amounts are optional and configured separately for each item within Item Code setup.

Note: For details on how to configure Item Codes and default values for them, refer to section 4 of the *System Setup User Manual*, available in the Transact Payments Support Library.

Although Item Code defaults secure most Checkout requests, they are not compatible with generic Item Codes with amounts that vary depending on what is selected for payment (for example, if you configured a Checkout for an athletic event with only a single Item Code and chose to send the description of the event in a Reference Type).

4.1.2 Hashing & Message Digests

For additional security, Transact Payments supports adding hashed message digests to Item Code amounts. Because you must send an amount in the message digest, however, you may not use it in conjunction with default amounts described above.

Transact Payments supports the following hashing and encoding schemes:

- MD5
- MD5 - Base64 Encoded
- SHA1
- SHA1 - Base64 Encoded
- HMAC-MD5
- HMAC-SHA1

Note: If you choose to use hashing and message digests, all of your requests must be authenticated before Transact Payments can perform payment processing. Requests that do not contain digests in your encoding scheme will be rejected.

Transact Payments will store the shared secret key for your hashing scheme. Upon receiving Checkout requests with message digests, Transact Payments will authenticate the request and only process the payment if authentication is successful.

4.1.2.1 Hashing Requirements

Depending on your encoding scheme, the hashing requirements may vary. However, you will need to fulfill the following requirements:

- Inform your Transact Consulting Team of the hashing scheme you want to use.
- Obtain the shared secret key from your Transact Consulting Team.
- Obtain the item amounts in clear text, formatted to 2 decimal points (for example, $99.99 would be `99.99`).
- Create the message digest based on the hashing scheme you want to use, for example:
  - MD5/SHA1 Hashing: Append the clear text amount to the key and then perform a MD5/SHA1 hash on the result.
  - Base64 Encoded Hashing: Append the clear text amount to the key, convert it to hex value, then perform the Base64 encoding, as described in the next section.
  - HMAC-MD5/HMAC-SHA1: Perform an HMAC hashing on the clear text amount, passing the key as an argument.
- Add the hashed value to the Checkout request using the digest parameter name.

Note: If a request contains multiple Item Codes, each one must have a separate, incrementing digest. For example, if the `digest` is your parameter name, the digest parameter for the first item would be `digest`; for the second item, `digest2`; for the third item, `digest3`; so on.

4.1.2.2 Example Hashing Process

This section lays out example steps to add a message digest to a request using MD5 - Base64 Encoded hashing. All the values provided below are examples and for information purposes only.

The example request, prior to adding the message digest, is:

```
https://commerce.cashnet.com/My_Checkout?itemcode=TUIT&amount=100.12
```

- As shown, the TUIT item amount, **formatted to 2 decimal places**, is `100.12`.
- In the example, the shared Checkout encryption key is `ENCRKEYTEST`.

To hash and add the digest:

1. Append the amount to the encryption key and obtain a value of `ENCRKEYTEST100.12`.
2. Hash the `ENCRKEYTEST100.12` value using MD5 to obtain a result of `469CE9F91E091B033033B20C851C371A`.
3. Convert the hex value to ASCII and perform a Base64 encoding to obtain a result of `Rpzp+R4JGwMwM7IMhRw3Gg==`.
4. In the Checkout request, after the amount for the associated Item Code, add the digest parameter and the message digest result to obtain the following value:

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=TUIT&
    amount=100.12&
    digest=Rpzp+R4JGwMwM7IMhRw3Gg==
```

4.2 Redirection via Sign-Out URL

Transact Payments 's sign-out URL option can be used to redirect customers after completing their payment in the Transact Payments application.

Note: Sign-out URL is also one of the options for Checkout Payment Notifications from Transact Payments (p. 15). You can use sign-out URL for only redirection, or for both redirection and payment notification.

Customers are transferred to a specified URL after making a payment in Transact Payments. The sign-out URL can contain the payment parameters you want to receive. You may configure the URL as a static field or send it as a parameter (`signouturl`) when transferring the customer to the Transact Payments application.

Transact Payments posts information back to the URL in standard query format as HTTPS GET or POST. You can configure separate URLs for successful and unsuccessful transactions.

4.3 Receipt Display Option

Once the user has successfully made a payment, you may choose to display a receipt to the user. The receipt can be used on its own or in conjunction with a redirect via Sign-Out URL. The details of the receipt are configured through the Transact Payments Administration site as part of the eMarket setup. This process is explained in the *eMarket Store Setup Manual*, available in the Transact Payments Support Library.

Note: To better ensure that the customer returns to your designated Sign-Out URL, you may want to skip the receipt page, because the customer will not need to manually select the Sign Out button to return to your application.

4.4 Checkout User Interface

The user interface for the Checkout screens can be customized to contain the logo, fonts, and text that you would like to display on the site. The details of the UI are configured through the Transact Payments Administration site as part of the eMarket setup. This process is explained in the *eMarket Store Setup Manual*, available in the Transact Payments Support Library.

4.5 API for Transaction Inquiries

Transact Payments can expose an application programming interface (API) for third-party applications to inquire about the status of a transaction by submitting a request with standard transactional information, such as the transaction number and associated Reference Types. The response from the Transact Payments API contains detailed transaction information in XML format.

Although the payment notification in Checkout integration is initiated by the Transact Payments application, you could potentially use web services to retrieve the payment status to verify that the payment was successful or use it to obtain other transactional data for your needs.

For more details on using the API for transaction inquiries, contact a Transact representative or refer to the *Transaction Inquiry Service Quick Guide*.

5 Reference Types

Transact Payments enables you to send additional information in the Checkout request, which can be used to facilitate in reporting, reconciling payment notifications, and identifying transactions. Common examples include the session ID and tax amount. This additional information must be sent in custom parameters, which are known in Transact Payments as *Reference Types*.

There are two main categories of Reference Types: *transaction-level* Reference Types and *item-level* Reference Types. The two types behave and are formatted differently; however, both types can be sent in payment notifications or responses from Transact Payments.

Notes:

- ✓ Although you can create Reference Types for any information that you want to send to Transact Payments, certain parameter names are hard coded into the Transact Payments system and cannot be used for your Reference Type names. For the list of these values, see Appendix A on page 31.
- ✓ For details on how to create and manage Reference Types in Transact Payments, refer to section 4.1 of the *System Setup User Manual*.

5.1 Transaction-Level Reference Types

Transaction-level Reference Types apply to the whole transaction. After creating transaction-level Reference Types in Transact Payments, they can be passed in for the Checkout. You may either associate them directly with the Merchant Code or leave them unassociated. If you associate Reference Types with the Merchant Code, they will be displayed for the customer to view and edit before completing the transaction. If you do not associate Reference Types with the Merchant Code, they will be passed to Transact Payments behind the scenes and can be configured to display on the receipt page.

Transaction-level Reference Types can also be used to pre-populate values users enter in your website in the Transact Payments Checkout, as described in Pre-Populating Checkout Form Parameters below.

Note: *Transaction-level* Reference Types are only available for eMarket Checkout and Storefront. If you are using a Gateway, this section does not apply to your integration.

5.1.1 Transaction-Level Reference Type Syntax & Example

Transaction-level Reference Types are formatted in a standard query name-value pair, as shown:

```
Reference Type Name=Reference Value
```

Below is an example GET request with two transaction-level Reference Types, SESSIONID and USERID. Note that you may send transaction-level Reference Types in any order in the request.

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=FEE&
  amount=123.00&
  SESSIONID=2334ffa&
  USERID=123446
```

5.1.2 Pre-Populating Checkout Form Parameters

Depending on the structure of your website or application, customers may be able to enter information in your web forms which could be used in payment processing in Transact Payments. Examples of such information include the customer's address, city, state, and email.

To minimize the need for your customers to re-enter payment information when they are transferred to the Checkout, you can include those details in your Checkout requests using transaction-level Reference Types. These details will then be used to pre-populate the Transact Payments Checkout screens.

Transact Payments comes pre-configured with certain Reference Types available for form pre-population, which you do not need to create or associate with your Merchant. One potential downside to using the pre-configured Reference Types, however, is that they apply across the whole Transact Payments system and could be difficult to manage and customize if you are using multiple stores. For the complete list of the pre-configured form parameters, see Appendix A (p. 31).

Alternatively, you may create any additional Reference Types for form pre-population. These Reference Types may be either global and apply to all stores like the default parameters, or they may be specific to a single store. You will configure each type of custom Reference Type differently in Transact Payments:

- **Global:** The Reference Type should end in _G like the pre-configured parameters and its settings should be applicable to all stores. All global parameters you create must be distinct from the default values listed in Appendix A (p. 31)
- **Store-Specific:** The Reference Type should begin with the Merchant Code and a hyphen (e.g., CHKT-) and its settings should be tailored for only one store.

Below you will find an example GET request that uses all of the default pre-population Reference Types (shown in red text) as well as two store-specific Reference Types (shown in green text).

Note: All form pre-population Reference Types function at the transaction level.

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=FEE&
  amount=75.00&
  NAME_G=John%20Doe&
  ADDR_G=525%20Oaks%20Drive&
  CITY_G=Springfield&
  STATE_G=PA&
  ZIP_G=123445&
  EMAIL_G=john%40doe.net&
  CHKT-DOB=09/01/1990&
  CHKT-LEVEL=Undergrad
```

5.2 Item-Level Reference Types

Item-level Reference Types are associated with items in the transaction and the Item Codes in Transact Payments.

Note: For details on how to create and manage Reference Types in Transact Payments, refer to section 4.1 of the *System Setup User Manual*.

5.2.1 Item-Level Reference Type Syntax

Item-level references are passed as **two name-value pairs**, which specify the Reference Type name and the value, as well as identify the Item Code with which the values are associated. Their syntax is as follows:

```
refYtypeX=Reference Type Name&
refYvalX=Reference Type Value
```

- X is a number that denotes the *Item Code* associated with the Reference Type. If there is only one Item Code, X must be **blank**. If there are multiple Item Codes in the request, they will be appended with numbers, starting at **2**. The X value maps to the Item Code's appended number.
- Y is a number that denotes the *Reference Type* associated with a particular Item Code. If there is only one Reference Type for the Item Code, Y must be **1**. If there are multiple Reference Types for an Item Code, the Y number must increment, the first Reference Type having a default value of **1**.

5.2.2 Item-Level Reference Type Examples

One item with one item-level Reference Type:

```
https://commerce.cashnet.com/SiteName?
  itemcode=APPFEE&
    amount=50.00&
    ref1type=SESSIONID&ref1val=45768
```

One item with two item-level Reference Types:

```
https://commerce.cashnet.com/SiteName?
  itemcode=APPFEE&
    amount=50.00&
    ref1type=SESSIONID&ref1val=45768&
    ref2type=USERID&ref2val=9900000
```

- As shown above, the digit following `ref`, or Y, increments because two Reference Types are associated with the same Item Code.
- Note that you can send the references in any order, provided the values are properly associated. Below is an example of the same Reference Types in a different order.

```
http://commerce.cashnet.com/SiteName?
  itemcode=APPFEE&
    amount=50.00&
    ref1type=USERID&ref1val=9900000&
    ref2type=SESSIONID&ref2val=45768
```

Two items with two item-level Reference Types for each item:

```
https://commerce.cashnet.com/SiteName?
  itemcode=APPFEE&
    amount=50.00&
    ref1type=SESSIONID&ref1val=45768&
    ref2type=USERID&ref2val=9900000&
  itemcode2=PARKING&
    amount2=150.00&
    ref1type2=LICENSE&ref1val2=bj45768&
    ref2type2=STATE&ref2val2=CA
```

- Note that the digit following `type`, or X, increments to identify the Item Code each set of Reference Types is associated with.

6 Gateway Integration Requirements

There are two main requirements for your third-party application to communicate with the eMarket Gateway: (1) you must send payment requests to the Transact Payments application containing the required transaction and payment information; and (2) you must have a method for handling synchronous payment responses from the Transact Payments system.

Notes:

- ✓ Transact Payments also provides a way for you to submit refund requests through your Gateway, explained in Gateway Transaction Refunds (p. 29).
- ✓ If you are using an eMarket Checkout, this section does not apply to your integration.

6.1 Gateway Payment Requests to Transact Payments

After a customer has selected items for purchase and entered payment information, your application will make an API call to the Transact Payments Gateway in order to process the payment.

6.1.1 Gateway Request Syntax

Your Gateway will be assigned a unique URL to communicate with Transact Payments. You will send Transact Payments transaction information in standard input parameters, formatted in query string name-value pairs separated by an ampersand (&).

The query string can be sent through HTTPS POST or GET. The Transact Payments application can parse your form input data. After doing so, the URL string will be formatted as follows:

HTTPS GET or POST String:

```
https://commerce.cashnet.com/GatewayName?
  name=value&
  name=value
```

Notes:

- ✓ In GET format, you must convert symbols into acceptable ASCII characters; accordingly, all GET examples in this guide will be properly encoded.
- ✓ In POST format, you do NOT need to convert symbols into ASCII characters; accordingly, all POST examples in this guide will NOT be encoded.

6.1.2 Required Gateway Request Information

Your Gateway request must contain the following basic information. Certain parameters used to send this information may be required and some may be optional.

- **Command:** The `command` parameter is required to indicate the type of transaction. There are two value options: `SALE`, for a sales transaction; and `REFUND`, for a refund of a transaction already created in Transact Payments. For standard payment request, you will need to send `command=SALE`. Refunds are explained in Gateway Transaction Refunds (p. 29).
- **Merchant:** Your Gateway will feature a unique identifier, known in Transact Payments as a Merchant ID. You must send this ID in the `merchant` parameter.
- **Station:** The `station` parameter is required to identify the Transact Payments credit card or ACH interface for processing the transaction and will be provided by your Transact Consulting Team.
- **Authentication credentials:** Gateway access is restricted by `operator` (aka username) and `password`, which are managed by Transact Payments and configured in Transact Payments. These credentials are required for all transactions.
- **Customer Code:** A Customer Code, such as a student ID, must be included in the `custcode` parameter. Alternatively, you may send a constant value for all transactions.
- **Shopping Cart Details:** You must send details about items selected to purchase. This will include the item codes and amounts, sent in the `itemcnt` (item count), `itemcode` and `amount` parameters, but may also include general ledger codes, quantity, and other parameters, depending on your preferences. See Appendix C Gateway Parameters (p. 42) for the complete list of available parameters.
- **Payment Tender Details:** You must send the payment tender information for the transaction, including, for example, the credit card number (`cardno`) and expiration date (`expdate`) for a credit card transaction. See Appendix C Gateway Parameters (p. 42) for the complete list of available parameters.

6.1.3 Detailed Gateway Syntax

Below is the standard syntax based on a request, containing the standard required request information. For the complete list of parameters that can be sent in the Gateway request as well as additional request examples, see Appendix C Gateway Parameters on page 42.

Request heading: all parameters listed are required:

```
https://commerce.cashnet.com/GatewayName?
  command=SALE&
  merchant=Merchant ID&
  station=Station ID&
  operator=Operator ID&
  password=Password&
  custcode=Student ID or constant value&
```

Shopping cart details: certain values are conditional:

```
  itemcnt=Number of items; ONLY send if more than 1&
  itemcode=First Item Code&
```

```
    amount=Amount associated with first Item Code – only used if you do NOT want to use
the default set amount in Transact Payments &
  itemcode#=Additional Item Code, if required, incrementing from 2&
    amount#=Amount associated with the Item Code with the appended #&
```

Note that you may also send `gl#`, `qty#`, and other parameters.

Payment tender details: certain values are conditional:

Credit card transactions require, at minimum, card number and expiration date; or the magnetic stripe data in the `cardno` parameter.

```
  cardno=Card Number&
  expdate=Card Expiration Date&
<!--OR-->
  cardono=Magnetic Stripe Data (from which Transact Payments will determine the exp.
date)
```

ACH transactions require, at minimum, a valid account number and routing number.

```
  acctno=Bank Account Number&
  rtno=9-Digit Routing Number
```

You may also send additional payment tender details for these transaction types, as well as payments for pin-less debit and campus card transactions.

6.1.4 Gateway Credit Card Example

Below is an example credit card transaction for two items.

```
https://commerce.cashnet.com/GatewayName?
  command=SALE&
  merchant=GTWAY&
  station=WEB&
  operator=GTWOPER&
  password=test1234&
  custcode=987654321&
  itemcnt=2&
  itemcode=REGFEE&
    amount=12.34&
  itemcode2=TICKET&
    amount2=25.00&
  cardno=Card Number&
  expdate=0319
```

The `itemcnt` parameter is included because there are two items; as a reminder, `itemcnt` is only required if there is more than one Item Code in the request.

Because the request contains multiple Item Codes, the first parameter is `itemcode`; the second, `itemcode2`. If there were a third, it would be `itemcode3`, and so forth. Furthermore, the associated `amount` parameters have the same appended number, to properly associate `amount` with `itemcode`, and amount2 with `itemcode2`.

6.1.5 Additional Gateway Request Information

Transact Payments enables you to send additional information in the Gateway request, which can be used to facilitate in reporting, reconciling payment notifications, and identifying transactions. Common examples include the session ID and citation number. This additional information must be sent in custom parameters, which are known in Transact Payments as *Reference Types*.

There are two main categories of Reference Types, *transaction-level* Reference Types, and *item-level* Reference Types. However, only item-level Reference Types can be used with an eMarket Gateway. For detailed information on item-level Reference Types, see Item-Level Reference Types (p. 23).

Reference Types can also be sent back in payment responses from Transact Payments.

6.2 Gateway Payment Responses from Transact Payments

In a Gateway, payment responses occur as a synchronous backend call to your application.

6.2.1 Gateway Response Syntax & Examples

Response messages from Transact Payments are formatted as a standard name-value pair query string wrapped in `<cngateway>` and `</cngateway>` HTML tags. f

6.2.1.1 Successful Transaction Syntax

Successful transactions are formatted according to the following syntax:

```
<cngateway>
  result=Constant of 0 to indicate success&
  busdate=Business Date in M/d/yyyy format&
  tx=Transact Payments transaction number
</cngateway>
```

6.2.1.2 Successful Transaction Example Response

```
<cngateway>
  result=0&
  tx=1234&
  busdate=6/25/2015&
</cngateway>
```

6.2.1.3 Failed Transaction Syntax

Failed transactions are formatted according to the following syntax. For the list of available non-zero results and their associated error messages, see Appendix D on (p. 46).

```
<cngateway>
  result=A non-zero digit associated with a specific error&
  failedtx=Transact Payments transaction number&
  respmessage=Error message associated with result number
</cngateway>
```

6.2.1.4 Failed Transaction Example Response

```
<cngateway>
  result=6&
  failedtx=1234&
  respmessage=Invalid credit card number or no credit card number provided
</cngateway>
```

6.3 Gateway Transaction Refunds

Transact Payments provides support for refunding payment transactions created in Transact Payments through the Gateway.

Upon receiving a refund request, Transact Payments reverses the electronic payment and creates a new transaction to track the refund and link to the original transaction. Credit card transactions are refunded to the card used for payment.

Notes:

- ✓ You may also process refunds through the *Find Transactions* tool in the Transact Payments Administration site. For details on how to do this, refer to the *Refunds and Dishonors Quick Guide* in the Transact Payments Support Library. Note that when refunding transactions through Transact Payments, you must manually refund the transaction in your campus's ERP system.
- ✓ You may only perform refunds for credit card and ACH transactions.
- ✓ The amount to be refunded must always be less than or equal to original transaction amount.

6.3.1 Gateway Refund Request Syntax

A refund transaction must contain a `command=REFUND` name-value pair as well as the standard `merchant`, `station`, `operator`, `password`, and `custcode` parameters.

In addition, you must specify the information about the refund using the `origtx`, `itemcode`, and `amount` parameters, as shown in the following request examples.

You will need to specify each Item Code included in the refund transaction separately, as well as the exact amount to refund for each Item Code. However, if you are refunding an Item Code that uses a default amount (p. 18), you may send no amount in the refund, and the default, total amount will be refunded.

If an item in the original transaction will not have any amount refunded, you can exclude that Item Code from the refund transaction. If an item will only be partially refunded, you can specify the partial amount to refund in the `amount` parameter.

Request heading: all parameters are required:

```
https://commerce.cashnet.com/GatewayName?
  command=REFUND&
  merchant=Merchant ID&
  station=Station ID&
  operator=Operator ID&
  password=Password&
  custcode=Student ID or constant value that must match that of the original
transaction&
```

Information about the refund; certain values are conditional:

```
  origtx=Original transaction number&
  itemcnt=Number of items; ONLY send if more than 1&
  itemcode=First Item Code&
    amount=Amount to be refunded for item in 2 decimal format e.g., 99.99&
  itemcode#=Additional Item Code, if required&
    amount#=Amount to be refunded for item in 2 decimal format e.g., 99.99
```

6.3.2 Gateway Refund Request Examples

Refunding credit card transaction with one item:

```
https://commerce.cashnet.com/GatewayName?
  command=REFUND&
  merchant=GTWAY&
  station=WEB&
  operator=GTWOPER&
  password=secret&
  custcode=987654321&
  origtx=12914&
  itemcode=REGFEE&
    amount=124.73
```

Refunding credit card transaction with two items:

```
https://commerce.cashnet.com/GatewayName?
  command=REFUND&
  merchant=GTWAY&
  station=WEB&
  operator=GTWOPER&
  password=secret&
  custcode=987654321&
  origtx=12914&
  itemcnt=2&
  itemcode=REGFEE&
    amount=124.73&
  itemcode2=APPFEE&
    amount2=25.00
```

Appendix A Checkout Requests

Checkout Request Parameters

Note: You may also send any Reference Types you have configured as well as the Form Pre-Population Parameters listed below.

PARAMETER NAME	REQUIRED, CONDITIONAL, OR OPTIONAL	DESCRIPTION
`virtual`	Conditional. Required for POST requests. Not used for GET.	Indicates the site name of the Checkout in POST requests.
`itemcode#`	Required.	Item Code for the product or service. For syntax details, see section 3.1.2 on page 14.
`amount#` or `transactionamount#`	Optional.	Amount to be charged for the item or service. If not provided, the amount will be taken from the Item Code table in Transact Payments. If provided, the Item Code amount set in Transact Payments will NOT be used.
`gl#`	Optional.	G/L code for the item or service. If not provided, the G/L will be taken from the Item Code table in Transact Payments. If provided, the Item Code G/L will NOT be used.
`qty#`	Optional.	Quantity for the item or service. Although you can pass quantities, Transact Payments will not calculate the total item amount based on quantity; therefore, you must pass the total amount for the item(s) in the amount parameter.
`desc#` or `transactiondesc#`	Optional.	Variable description of the Item Code.
`refYtypeX`	Optional.	First name-value pair for a Reference Type, defining the parameter name. For syntax details, see section 5.2.1 on page 23.
`refYvalX`	Optional.	Second name-value pair for a Reference Type, defining the value associated with the parameter. For syntax details, see section 5.2.1 on page 23.
`digest#`	Conditional. If you choose to send digests, you must always send digests.	Defines the hashed amount for an item or service. For syntax details, see section 4.1.2 on page 18. Incompatible with default Item Code amounts. You may change the “digest” value in the Merchant in Transact Payments.
`custcode`, `eusername`, `ousername` or `username`	Optional.	Transact Payments customer ID of customer. A default value may be sent as well.
`lname`	Optional.	Customer’s last name.
`fname`	Optional.	Customer’s first name.
`city`	Optional.	Customer’s billing city.
`state`	Optional.	Customer’s billing state.

PARAMETER NAME	REQUIRED, CONDITIONAL, OR OPTIONAL	DESCRIPTION
zip	Optional	Customer's billing zip code.
country	Optional	Customer's billing country.
addr	Optional	Customer's billing address.
shipaddr_as_bill_addr	Optional	Indicates whether or not the customer's shipping address is the same as the billing address.
email	Optional	Customer's email address.
signouturl	Optional	Indicates the URL to which customers will be directed after a successful transaction. For example: signouturl=https://studyabroad.university.edu/index.cfm?FuseAction=Gateways.Success
incompletesignouturl	Optional	Indicates the URL to which customers will be directed after a failed transaction, enabling them to complete payment. For example: incompletesignouturl=https://university.destinysolutions.com/srs/enrolmgr/common/account/processExternalPayment.do?
acctname	Optional	Can be used to pre-populate the account name on the credit card display in the Checkout page.
<ref_type>_EDT	Optional	Can be used to indicate whether the corresponding transaction-level Reference Type is editable by the user in the Checkout interface. Y or N may be sent as values. Y will make the Reference Type value editable. N will make it non-editable. Null will be ignored. For example: NAME_G=John Doe&NAME_G_EDT=N
edit_indicator	Optional	Can be used to disable editing of all Item Code amounts for specific eMarket Checkout customers, while still allowing other customers to do so. If sent, the value can be Y, N, or null; however, only N makes a change to the customer's checkout process: • N will not allow the customer to edit any Item Code amounts in the checkout transaction, regardless of any Transact Payments settings, including the Merchant Code's Checkout Allows Partial Payments (Checkout Only) setting and all of the Item Codes' Allow Amount to be Changed settings. • Y and null will keep Item Codes amounts editable where allowed at the merchant level (in the Merchant Code's Checkout Allows Partial Payments (Checkout Only) setting) and at the item level (in the Item Code's Allow Amount to be Changed setting).
pmt_indicator	Optional	Can be used to restrict the transaction's payment method to only one available method for the customer. The following values may be sent: • A – Only the merchant's ACH Payment Code will be allowed. • S – Only the merchant's ACH Payment Code will be allowed. • C – Only the merchant's credit card Payment Code will be allowed. • I – Only the merchant's IFT Payment Code will be allowed. • NULL or blank – Any Payment Code available in the merchant will be allowed (same behavior as not sending the parameter).

PARAMETER NAME	REQUIRED, CONDITIONAL, OR OPTIONAL	DESCRIPTION
alt_otpsso_url	Optional	Provides the option to dynamically define an alternate notification-only URL. When included, this URL will be used for notification and will override the URL information specified in the Store Setup HTTP Notifications section.

Default Form Pre-Population Reference Types

PARAMETER NAME	DESCRIPTION
NAME_G	Customer's full name as shown on payment card.
ADDR_G	Customer's billing address.
CITY_G	Customer's billing city.
STATE_G	Customer's billing state.
ZIP_G	Customer's billing zip code.
EMAIL_G	Customer's email address.

Transact Payments Keywords that Cannot Be Used as References

In addition to **all the Transact Payments Request Parameters and Default Form Repopulation Reference Types listed above**, you may not send the following parameter names as Reference Types.

AUTHMETHOD	CHECKOUTDIGESTPARAM (prefix)
CLIENT	LT
PV	SID
SUBMIT	TICKET

Checkout Request Examples

One item (GET request):

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=APPFEE&amount=50.00
```

Three items (POST request):

```
https://commerce.cashnet.com/404Handler/pageredirpost.aspx?
  virtual=CheckoutName&
  itemcode=APPFEE&amount=50.00&
  itemcode2=PARKING&amount2=150.00&
  itemcode3=TRANSCRIPT&amount3=50.00
```

One item with one item-level reference type (GET request):

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=APPFEE&amount=50.00&
    ref1type=SESSIONID&ref1val=45768
```

One item with two item-level reference types (POST request):

```
https://commerce.cashnet.com/404Handler/pageredirpost.aspx?
  virtual=CheckoutName &
  itemcode=APPFEE&amount=50.00&
    ref1type=SESSIONID&ref1val=45768&
    ref2type=USERID&ref2val=9900000
```

Two items with two item-level reference types for each item (GET request):

```
https://commerce.cashnet.com/CheckoutName?
  itemcode=APPFEE&amount=50.00&
    ref1type=SESSIONID&ref1val=45768&
    ref2type=USERID&ref2val=9900000&
  itemcode2=PARKING&amount2=150.00&
    ref1type2=LICENSE&ref1val2=bj45768&
    ref2type2=STATE&ref2val2=CA
```

One item with one transaction-level reference type (POST request):

```
https://commerce.cashnet.com/404Handler/pageredirpost.aspx?
  virtual=CheckoutName&
  itemcode=APPFEE&amount=50.00&
  SESSIONID=66666
```

Appendix B Checkout HTTP Notifications

This section contains the parameters that can be sent in Checkout responses using the HTTP Notification and Sign-Out URL payment notification methods, as well as example HTTP notifications.

For other notification methods, notifications can be in the format you prefer and use a flexible parameter structure that suits your needs.

HTTP Notification & Sign-Out URL Parameters

Note: Any Reference Types you send in your Checkout requests will also be sent in notifications.

PARAMETER NAME	DESCRIPTION	ALWAYS SENT?	SUCCESS OR FAILURE MESSAGES, OR BOTH?
`result`	Transaction result. The digit zero (0) indicates success. All other numerical responses indicate a failure. See HTTP Notification Error Codes below for the list of failure notifications associated with result numbers.	Y	Both
`respmessage`	Status description for the transaction. Either SUCCESS or a detailed failure notification, as listed in HTTP Notification Error Codes below.	Y	Both
`ccerrorcode`	The credit card error code, if the payment type is credit card and there is a credit card processing error. See HTTP Notification Error Codes below for the list of cc error codes and associated error messages.	N	Failure
`ccerrormessage`	The error message associated with the credit card error code, as listed in HTTP Notification Error Codes below.	N	Failure
`merchant`	Name of the Transact Payments Merchant. This is a unique code to identify the Transact Payments Checkout.	Y	Both
`custcode`	Transact Payments identified customer code for the transaction. If the eMarket Checkout does not require login, this will be the default customer code configured in Transact Payments for this particular Merchant.	Y	Both
`operator`	Transact Payments Operator ID (aka, username) configured for the Checkout.	Y	Both
`station`	The Transact Payments station, used to identify the credit card or ACH interface for processing the transaction.	Y	Both
`batchno`	The Transact Payments batch number associated with the transaction.	Y	Both
`lname`	Last name of the customer who made the payment. If the eMarket Checkout does not require login, this will be the identity associated with the default Customer Code configured in Transact Payments for the Merchant.	N	Both

PARAMETER NAME	DESCRIPTION	ALWAYS SENT?	SUCCESS OR FAILURE MESSAGES, OR BOTH?
fname	First name of the customer who made the payment. If the eMarket Checkout does not require login, this will be the identity associated with the default Customer Code configured in Transact Payments for the Merchant.	N	Both
tx	Transact Payments transaction number if the transaction is a success.	N	Success
failedtx	Transact Payments transaction number if the transaction is a failure.	N	Failure
pmtcode	Transact Payments Payment Code for the transaction	N	Both
cctype	If a credit card transaction, the credit card type. M stands for MasterCard, V for Visa, A for American Express, and D for Discover.	N	Both
effdate	Effective date for the transaction.	N	Both
addr	Customer's billing address.	N	Both
addr2	Second billing address line.	N	Both
addr3	Third billing address line.	N	Both
city	Customer's billing city.	N	Both
state	Customer's billing state.	N	Both
zip	Customer's billing ZIP code.	N	Both
email	Customer's email address.	N	Both
shippingaddr1	Customer's shipping address.	N	Both
shippingaddr2	Second shipping address line.	N	Both
shippingaddr3	Third shipping address line.	N	Both
shippingcity	Customer's shipping city.	N	Both
shippingstate	Customer's shipping state.	N	Both
shippingzip	Customer's shipping ZIP code.	N	Both
txstatus	Transaction status for the Flywire transaction. Status options include Pending (P), Closed (C), and Voided or Canceled (X).	N	Both
itemcnt	Indicates the number of item codes passed in the Gateway request.	N	Both
itemcode#	Item Code for the product or service. If more than one, Item Codes increment (i.e., itemcode2 for the second Item Code parameter).	Y	Both
amount#	Amount charged for each particular item as determined by the appended number, if applicable.	Y	Both
qty#	Quantity for a particular item as determined by the appended number, if applicable.	N	Both

PARAMETER NAME	DESCRIPTION	ALWAYS SENT?	SUCCESS OR FAILURE MESSAGES, OR BOTH?
gl#	G/L Code for a particular item as determined by the appended number, if applicable.	N	Both
refYtypeX	First name-value pair for a Reference Type, defining the parameter name. For syntax details, see section 5.2.1 on page 23.	N	Both
refYvalX	Second name-value pair for a Reference Type, defining the value associated with the parameter. For syntax details, see section 5.2.1 on page 23.	N	Both

HTTP Notification Examples

Successful credit card transaction for one item:

```
result=0&
respmessage=SUCCESS&
merchant=CHKT&
custcode=987654321&
operator=WEB&
station=WEB&
pmtcode=CC&
lname=Doe&
fname=Joe&
addr=525 Oaks Drive&
batchno=6&
tx=43&
cctype=M&
email=john@doe.com&
effdate=1/1/2015&
itemcode=REGFEE&
  amount=114.00&
  qty=1&
  gl=000000
```

Successful ACH transaction for one item:

```
result=0&
respmessage=SUCCESS&
merchant=CHKT&
custcode=987654321&
operator=WEB&
station= WEB&
pmtcode=ACHWEB&
lname=Doe&
fname=Jane&
```

```
addr=145 Edgewood Ct.&
batchno=198&
tx=12&
email=janedoe@domain.com&
effdate=12/19/2015&
itemcode=REGFEE&
  amount=99.00&
  qty=1&
  gl=000000
```

Successful credit card transaction for two items and two item-level references each:

```
result=0&
respmessage=SUCCESS&
merchant=CHKT&
custcode=987654321&
operator=CHECKOUT&
station=WEB&
pmtcode=CC&
lname=Doe&
fname=John&
addr=11 Maple Ave.&
batchno=125&
tx=1998&
cctype=V&
email=jd@school.edu&
effdate=8/11/2014&
itemcnt=2&
itemcode=REGFEE&
  amount=10.00&
  qty=1&
  gl=000000&
  ref1type=REGNO&
    ref1val=3rty345&
  ref2type=REGYEAR&
    ref2val=2008&
itemcode2=APPFEE&
  amount2=100.00&
  qty2=1&
  gl2=000000&
  ref1type2=APPNO&
    ref1val2=666666&
  ref2type2=APPYEAR&
    ref2val2=2009
```

Failed credit card transaction for one item:

```
  result=230&
  respmessage=DECLINE. Your credit card has been declined. If you have questions about
the decline, please call the number on the back of your credit card&
  ccerrorcode=230&
  ccerrormessage=DECLINE. Your credit card has been declined. If you have questions
about the decline, please call the number on the back of your credit card&
  merchant=CHKT&
  custcode=987654321&
  operator=WEB&
  station=WEB&
  pmtcode=ACHWEB&
  lname=Doe&
  fname=John&
  addr=101 Riesling Dr&
  batchno=4&
  tx=25&
  cctype=M&
  email=doe@hopkin.edu&
  effdate=9/23/2014&
  itemcode=REGFEE&
    amount=55.00&
    qty=1&
    gl=000000
```

Failed ACH transaction for one item:

```
  result=13&
  respmessage=Unable to Process ACH since Account Number or Routing Transit Number is
missing.&
  merchant=CHKT&
  custcode=987654321&
  operator=WEB&
  station=WEB&
  pmtcode=ACHWEB&
  lname=Roberts&
  fname=Eric&
  addr=95 Nor Way&
  batchno=19&
  tx=2550&
  cctype=AMEX&
  email=ered@domain.com&
  effdate=6/3/2015&
  itemcode=REGFEE&
    amount=45.00&
    qty=1&
    gl=000000
```

HTTP Notification Error Codes

Error codes are identical for both the `result` and the `ccerrorcode` parameters and the associated messages in the `respmessage` and `ccerrorcode` parameters. Credit card error codes and messages, however, are only sent for failed credit card transactions.

RESULT / CC ERROR CODE	RESPONSE MESSAGE / CC ERROR MESSAGE
1	Invalid customer code or no customer code specified
4	An invalid item code has been supplied
5	Negative amount is not allowed
6	Invalid credit card number or no credit card number provided
7	Invalid expiration date or no expiration date provided
9	Invalid ACH account number or no account number provided
10	Invalid routing/transit number or no routing/transit number provided
11	Invalid account type or no account type provided
12	Invalid check digit for routing/transit number
13	Unable to Process ACH since Account Number or Routing Transit Number is missing.
21	Invalid merchant code or no merchant code has been supplied
22	Invalid client code or no client code has been supplied
25	Invalid amount or amount not provided
230	DECLINE. Your credit card has been declined. If you have questions about the decline, please call the number on the back of your credit card
231	REFERRAL. Your credit card has been declined. If you have questions about the decline, please call the number on the back of your credit card
232	AVS FAIL. For security reasons, we cannot accept your credit card
233	CID FAIL. For security reasons, we cannot accept your credit card
234	OTHER ERROR. Your credit card has been declined. If you have questions about the decline, please call the number on the back of your credit card
235	OTHER INTERNAL ERROR. An error occurred while processing your credit card. Your card has not been charged
701	This website has been disabled
702	Improper merchant code. Please contact the system administrator. (not a checkout type of merchant)
703	This site is temporarily down for maintenance. We regret the inconvenience. Please try again later. **Note:** This value can be customized in Store Setup's Custom Messages in the Site Offline Message.
704	Duplicate item violation
705	An invalid reference type has been passed into the system
706	Items violating unique selection have been passed in
801	Invalid Campus Card Number

RESULT / CC ERROR CODE	RESPONSE MESSAGE / CC ERROR MESSAGE
802	Campus Card has insufficient funds
803	Campus Card processing failure
901	Customer Cancelled before processing payment (from fee notice)
902	Customer Cancelled before processing payment

Appendix C Gateway Requests

Gateway Request Parameters

PARAMETER NAME	REQUIRED, CONDITIONAL, OR OPTIONAL	DESCRIPTION
`command` or `resource`	Required	Indicates the type of transaction. `SALE` is used for a sales transaction and `REFUND` for a refund transaction.
`merchant`	Required	Unique Merchant Code used to identify your eMarket Gateway.
`station`	Required	Transact Payments Station ID configured for the Gateway.
`operator`	Required	Transact Payments Operator ID configured for the Gateway.
`password`	Required	Password for the Gateway Operator ID.
`custcode`	Required	Transact Payments customer ID of customer. A default value may be used as well; however, you must send it either way.
`itemcnt`	Conditional. Required ONLY when there is more than one Item Code.	Indicates the number of Item Codes passed in the Gateway request. If only one, do NOT send this parameter.
`itemcode#`	Required	Item Code for the product or service. For syntax details, see section 3.1.2 on page 14.
`amount#`	Optional	Amount to be charged for the item or service. If not provided, the amount will be taken from the Item Code table in Transact Payments. If provided, the Item Code amount set in Transact Payments will NOT be used.
`gl#`	Optional	G/L code for the item or service. If not provided, the G/L will be taken from the Item Code table in Transact Payments. If provided, the Item Code G/L will NOT be used.
`qty#`	Optional	Quantity for the item or service. Although you can pass quantities, Transact Payments will not calculate the total item amount based on quantity; therefore, you must pass the total amount for the item(s) in the amount parameter.
`refYtypeX`	Optional.	First name-value pair for a Reference Type, defining the parameter name. For syntax details, see section 5.2.1 on page 23.
`refYvalX`	Optional.	Second name-value pair for a Reference Type, defining the value associated with the parameter. For syntax details, see section 5.2.1 on page 23.
`cardno`	Conditional. Only required for credit card transactions	The credit card number. Must contain only digits without any spaces or dashes. You may alternatively send be the magstripe data in this parameter.
`expdate`	Conditional. Only required for credit card transactions.	The credit card expiration date in MMYY format. Only send if you are NOT sending the magstripe date in the `cardno` parameter.
`cid`	Optional.	The Customer Identification Number (CID) associated with the credit card.

PARAMETER NAME	REQUIRED, CONDITIONAL, OR OPTIONAL	DESCRIPTION
acctno	Conditional. Only required for ACH transactions.	The customer's bank account number.
rtno	Conditional. Only required for ACH Transactions.	The 9-digit bank routing number.
accttype	Optional.	Indicates the type of account use for ACH transactions. C can be used for Checking and S for savings. If no value is sent, Transact Payments will assume it is a checking account.
acctname	Optional.	Indicates the name of the bank account.
pldno	Conditional. Only required for pin-less debit transactions.	Pin-less debit number.
campuscard	Conditional. Only required for campus card transactions.	Campus card number.
pmtcode	Optional.	Payment code associated with transaction. Transact Payments uses logic to determine this based on your shopping cart details; however, you may manually send it, provided it matches your Transact Payments Payment Code exactly.
lname	Optional.	Customer's last name. Only relevant if the student's distinct Customer Code is sent in the CUSTCODE. If you use a default Customer Code, then use a Reference Type to send this information.
fname	Optional.	Customer's first name. Only relevant if the student's distinct Customer Code is sent in the CUSTCODE. If you use a default Customer Code, then use a Reference Type to send this information.
city	Optional.	Customer's billing city.
state	Optional.	Customer's billing state.
zip	Optional.	Customer's billing ZIP code.
addr	Optional.	Customer's billing address.
origtx	Conditional. Refund Only.	Original Transact Payments transaction number.
campuscard	Conditional. Only required for campus card transactions.	Campus card number.

Gateway Request Examples

One item, paying with credit card:

```
https://commerce.cashnet.com/GatewayName?
  command=SALE&
  merchant=GTWAY&
  station=WEB&
  operator=GTWOPER&
  password=test1234&
  custcode=987654321&
  itemcode=REGFEE&
    amount=12.34&
  cardno=Card Number&
  expdate=0319
```

One item, paying with electronic check:

```
https://commerce.cashnet.com/GatewayName?
  command=SALE&
  merchant=GTWAY&
  station=WEB&
  operator=GTWOPER&
  password=test1234&
  custcode=987654321&
  itemcode=REGFEE&
    amount=12.34&
  acctno=Account Number&
  rtno=121000056&
  accttype=C
```

Two items with one item-level Reference Type each, paying with credit card:

```
https://commerce.cashnet.com/GatewayName?
  command=SALE&
  merchant=GTWAY&
  station=WEB&
  operator=GTWOPER&
  password=test1234&
  custcode=987654321&
  itemcnt=2&
  itemcode=REGFEE&
    amount=12.34&
    ref1type=REGNO&
      ref1val=APQ23
  itemcode2=CITATION&
    amount2=20.00&
    ref1type2=CITNO&
```

```
      ref1val2=23221
  cardno=Card Number&
  expdate=0319&
```

Three items with two item-level Reference Types each, paying with credit card:

```
https://commerce.cashnet.com/GatewayName?
  command=SALE&
  merchant=GTWAY&
  station=WEB&
  operator=GTWOPER&
  password=test1234&
  custcode=987654321&
  itemcnt=3&
  itemcode=REGFEE&
    amount=12.34&
      ref1type=REGNO&
        ref1val=APQ23&
      ref2type=EVENTNO&
        ref2val=234&
  itemcode2=CITATION&
    amount2=20.00&
      ref1type2=LICENSENO&
        ref1val2=CA1234&
      ref2type2=CITNO&
        ref2val2=23221
  itemcode3=APPFEE&
    amount3=25.00&
      ref1type3=APPNO&
        ref1val3=AP335&
      ref2type3=SESSIONID&
        ref2val3=342222&
  cardno=Card Number&
  expdate=0319
```

Appendix D Gateway Responses

Gateway Response Parameters

RETURN	DESCRIPTION
`result`	Transaction result. The digit zero (0) indicates success. All other numerical responses indicate a failure. See the section below for the list of failure notifications associated with result numbers.
`tx`	Transaction (or receipt) number if the transaction is a success. This parameter is NOT sent for failed transactions.
`busdate`	Business Date of a successful transaction in **M/d/yyyy format**. This parameter is NOT sent for failed transactions.
`failedtx`	Transaction number for a failed transaction. This parameter is NOT sent for successful transactions.
`respmessage`	Status description for a failed transaction describing the cause of the failure, as listed in Gateway Response Message Error Codes below. This parameter is NOT sent for successful transactions.
`ccerrorcode`	Credit card processing error if there is a credit card error. Only sent for failed credit card transactions.
`ccerrormessage`	For each credit card error code, there is a specific message describing the error. For the list of errors, see Gateway Credit Card Error Codes below.

Gateway Response Message Error Codes

RESULT	RESPONSE MESSAGE
1	Invalid customer code or no customer code specified
2	Invalid operator code or no operator specified
3	Invalid workstation code or no station specified
4	Invalid item code or no code specified
5	Negative amount is not allowed
6	Invalid credit card number or no credit card number provided
7	Invalid expiration date or no expiration date provided
8	Please only provide either ACH, PIN-less or credit card information
9	Invalid ACH account number or no account number provided
10	Invalid routing/transit number or no routing/transit number provided
11	Invalid account type or no account type provided
12	Invalid check digit for routing/transit number
13	No ACH merchant account setup for the location of the station being used

RESULT	RESPONSE MESSAGE
14	Invalid ACH account holder name or no account holder name provided
21	Invalid merchant code or no merchant code provided
22	Invalid client code or no client code provided
23	Invalid password or no password provided
24	Invalid transaction type or no transaction type provided
25	Invalid amount or amount not provided
26	Invalid payment code provided
27	Invalid version number or version not found
31	Application amount exceeds account balance
150	Invalid payment information or no payment information provided
200	Invalid command
201	Customer not on file
205	Invalid operator or password
206	Operator is not authorized for this function
208	Customer/PIN authentication unsuccessful
209	Credit card error
211	Credit card error
212	Customer/PIN not on file
213	Customer information not on file
215	Old PIN does not validate
221	Invalid credit card processor type specified in location or payment code
222	Credit card processor error
280	SmartPay transaction not posted
301	Original transaction not found for this customer
302	Amount to refund exceeds original payment amount or is missing
304	Original credit card payment not found or corrupted
305	Refund amounts should be expressed as positive amounts
306	Original ACH payment not found
307	Original electronic payment not found
308	Invalid original transaction number or original transaction number not found
310	Refund amount exceeds amount still available for refund
311	Total charges that can be refunded is $0.00
321	Store has not been implemented
501	Unable to roll over batch

RESULT	RESPONSE MESSAGE
502	Batch not found
503	Batch information not available
650	Invalid quick code
651	Transaction amount does not match amount specified in quick code
652	Invalid item code in the detail of the quick code
901	Plan number not defined
902	Customer does not have a campus card number in the system
903	Order does not exist
904	Order was already fulfilled
905	Unable to fulfill order
906	Because of the length of time that has elapsed since the original credit card authorization, payment for transaction [TX NO] required a re-authorization, which was not successful. The order cannot be fulfilled at this time.
907	Insufficient parameters. A transaction number or reference data is required.
908	Unable to perform inquiry.
909	Online Lookup failed.
910	Other Failure.
911	Customer not found
-1	NO RESULT
-2	Invalid request parameter(s).
999	An unexpected error has occurred. Please consult the event log.
1010	Invalid refundpreference or no refundpreference specified.
1011	SmartPay transactions may not be processed through the transaction gateway.
1012	Fulfillment is not allowed for this IFT transaction.
Anything Else	An error has occurred

Gateway Credit Card Error Codes

CC	CC ERROR MESSAGE
230	CNCreditCardProcessor.CCErrorType.decline
231	CNCreditCardProcessor.CCErrorType.referral
232	CNCreditCardProcessor.CCErrorType.avs_fail
233	CNCreditCardProcessor.CCErrorType.cid_fail
234	CNCreditCardProcessor.CCErrorType.other_processor_error
235	CNCreditCardProcessor.CCErrorType.other_internal_error