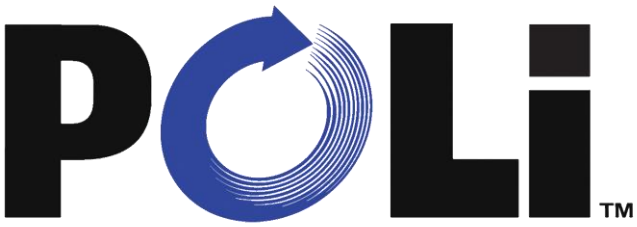


Merchant Implementation Guide

Version 5.0

Disclaimer and limitation of Liability

This Merchant Implementation Guide sets out the proper procedures for the operation and use of the POLi™ service offered by Centricom Pty Limited. This service, and your access to them, is governed by the terms and conditions of the Centricom Merchant Agreement. Only persons who have signed and agreed to be bound by the Centricom Merchant Agreement may use the transaction processing services offered by Centricom Pty Limited (including POLi™).

Trademarks

Centricom, the Centricom Logo, POLi™, and other trademarks, service marks, and logos are registered or unregistered trademarks of Centricom Pty Limited in Australia and in foreign countries. Other trademarks and service marks in this document are the property of their respective owners.

Table of Contents

Executive Summary / Abstract

POLi™ enables customers to use their internet banking facility to make payments direct to a merchant's nominated account.

A set of web services – referred to herein as POLi™ Merchant API – is provided so that merchants can integrate POLi™ into their e-commerce website. Integration with the POLi™ Merchant API allows merchants to initiate POLi™ payment requests, make real-time transaction status enquiries and receive notification of successful or unsuccessful payments for goods and services sold.

Aside from the technical integration of the API into the e-commerce website, the merchant must meet a series of contractual and operational obligations as part of the merchant implementation process. The administrative tasks involved in merchant implementation include providing Centricom with a complete Customer Information Record (CIR) which includes the company details, key contacts, and provision of at least one bank account into which payments are to be deposited. Additionally, execution of relevant legal documentation and contracts is also required.

Once the CIR has been received and processed at Centricom, the merchant will be provided with the relevant authentication details to successfully interface and communicate with POLi™. In order to facilitate access to ongoing merchant support services offered through the administration and reporting portal known as the 'POLi™ Console', a Business Manager username and password will also be provided.

This document describes in detail the requirements for technical integration and is intended for use by a cross section of merchant staff, including technical developers and business operations personnel.

Preface

NOTATIONAL CONVENTIONS

Typeface	Application
`Courier New`	Font Size 10. Field names for data submitted in a web service request or response
`Consolas`	Font size 9. Sample code for REST web methods

ICONS

Where you see the following icons, please take note of their meaning:


Indicates best practice


Indicates information that should be taken into consideration


Indicates a warning or risk

DOCUMENTATION ERRORS

If you find any errors please e-mail implementation@polipayments.com with “Documentation” in the subject line. Please provide a detailed description of the error including page number, section and description.

RELATED DOCUMENTS

Name	Description
POLi™ Console User Guide for Merchants	A walkthrough of the POLi™ Console including reporting and merchant maintenance.
POLi™ Merchant Style Guide	A guide to the stylistic requirements of a POLi™ implementation on a merchant website. http://www.polipayments.com/assets/docs/MerchantStyleGuide.pdf

Terms and Definitions

Term	Definition
Buffer	A 30 second allowance for communications deducted from the merchant threshold to determine the POLi™ timeout.
Customer	An online buyer that uses POLi™ to pay for their purchase on the merchant's website. POLi™ is only suitable for personal payments to a merchant, so we also refer to the customer as the consumer.
iBank	A fictitious internet bank site operated by POLi where merchants can complete test transactions that have no monetary value. iBank is designed to simulate the behaviour of real internet banking services.
Merchant	An online seller that offers POLi™ as a payment option.
Merchant Group	A collection of merchants operating under a parent group.
VAR	A Value Added Reseller who maintains and operates the POLi™ service for one or more groups of merchants.
Merchant Threshold	The time after which the merchant will not be able to proceed with a customer's transaction.
POLi™ Console	An online resource where merchants can run reports and maintain their details.
Nudge	HTTP Post sent by the POLi™ host to a nominated URL to inform the merchant that the POLi™ transaction has reached a terminal status. It indicates that the merchant should perform the GetTransaction web service to enquire the results of the transaction.
Timeout	The time in seconds after a transaction is initiated by the Merchant that the transaction will become unable to be completed. Passed to the POLi™ in the InitiateTransaction web service and/or stored in the merchant's POLi™ configuration.
Token	Encrypted time-stamped data related to a POLi ID that is only valid for the life of a transaction and is used for identification of a transaction and data related to it.
POLi ID	A unique reference for each POLi™ transaction used for reconciliation and tracking. The ID is assigned by POLi™ when the transaction is initiated. It appears on bank statements and POLi™ reports.
WSDL	Web Services Description Language is an XML-based language for describing web services and how to access them.

1 What is POLi™?

POLi™ is an online payment service that enables customers to safely pay registered online merchants directly from their own Internet banking facility. POLi™ guides a user through the internet banking transaction process and notifies the merchant when a transaction is completed.

1.1 How does POLi™ work?

When a POLi™ transaction is successfully initiated by a merchant, the purchasing customer is redirected to a series of web pages hosted by POLi™. They then select the bank they wish to use to make the payment and are taken to their bank's Internet Banking interface.

The customer is then prompted to log into their Internet banking site where they are then guided through a 3rd party payment.

Upon reaching the 3rd party payment page, POLi™ pre-populates the merchant's registered bank account details, the purchase amount and purchase related reference data into the applicable fields served within the internet banking site. The customer then confirms all details and completes the payment. The conclusion of the 3rd party payment process is validated by POLi™ and receipt information is retained for reconciliation and reporting back to the merchant.

The user is then redirected back to the merchant site where the merchant can generate a purchase receipt and complete the transaction at the merchant site. If a merchant provides a notification URL to POLi™, POLi™ will additionally send confirmation to the merchant notifying them that the transaction has been completed (via an HTTP Post referred to as a 'POLi™ Nudge'). The POLi™ Nudge signifies to the merchant that they can now retrieve the full results of the transaction.

1.2 How do Merchants communicate with POLi™?

POLi™ exposes a set of web services known as 'POLi™ Merchant API' that can be called by merchants to:

- Initiate a transaction,
- Query the details and status of transactions
- Obtain a list of banks that can be used with POLi™ and
- To generate a Payment URL.

These web services can be called either by using the SOAP protocol or by posting XML data (REST) to the POLi™ server.

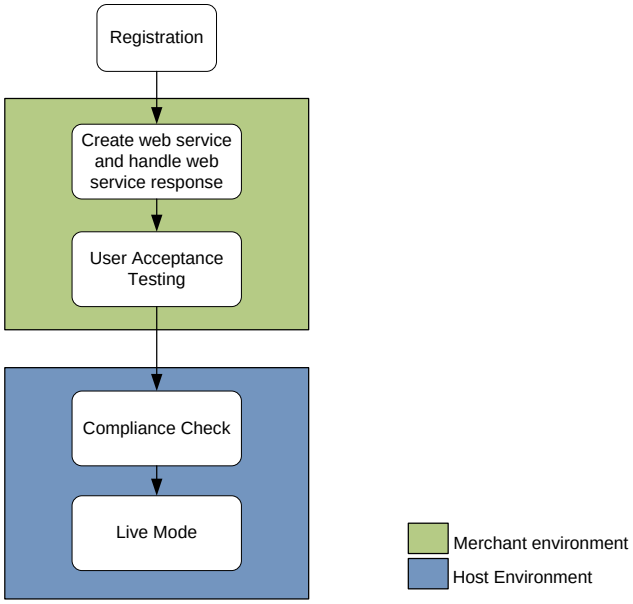
2 Merchant Implementation Process

In order to make POLi™ transactions, merchants must complete a Customer Information Record and have finalised all legal and contractual obligations.

Upon completion of this process, a POLi™ Implementation Manager will create an account in the POLi™ system and provide details on how to access the POLi™ Console. All merchants must review the account details created by logging in to the POLi™ Console and confirming the POLi™ Configuration.

Note: For more information on the POLi™ Console, please refer to the POLi™ Console User Guide for Merchants.

The Merchant Implementation Process for POLi™ involves 4 phases:



1/ Registration with your POLi™ Product Manager	Creation of a merchant account with POLi. Your POLi™ Implementation Manager should supply you with a Merchant Code and Authentication Code. The merchant must provide POLi with information such as the bank account to be credited for POLi™ transactions and contact and billing details.
2/ Technical Integration of POLi™ into merchant checkout process	Integrate POLi™ to your payment page by calling the appropriate POLi™ web services with the correct parameters. Ideally, this first implementation should be to a development or test system, rather than the production environment.
3/ Merchant Acceptance Testing	Ensure POLi™ is operational and resolve any functional or usability issues.
4/ Compliance Check	POLi Payments ensures that the implementation of POLi™ meets contractual and functional standards before moving to Live mode.

2.1 Implementation Considerations

To ensure an on-time, high-quality implementation of the POLi™ service as a payment option into your website, it is paramount that implementation and future operations be well considered.

To assist the planning process, a short list of important things to consider has been compiled.

- Timeouts
- Merchant Reference Field
- Merchant Data
- Database Design and Data Storage
- Customer Support Details
- Use of Merchant URLs
- POLi™ Receipt

2.1.1 Timeouts

POLi™ enforces the notion of a 'timeout', that is, the length of time within which a POLi™ transaction must be completed. Using the POLi™ Console, merchants can specify a default timeout value to apply to all transactions submitted. In addition, a merchant can specify a custom timeout value for a specific transaction when initiating that transaction. If this value is supplied, it will supersede the default value set up in the POLi™ Console. This may better facilitate purchases that are time sensitive. It is recommended that merchants allow a 30-seconds communications buffer when setting their POLi™ timeout value.

Note: The Timeout value should be noted and considered in assigning the timing of the transaction status enquiries.

Note. Merchant timeout value must be greater than POLi™ timeout value. This is to prevent POLi™ transaction completing only to find the transaction by the merchant has expired.'

2.1.2 Merchant Reference Field

The merchant reference field is an important piece of information used predominately for reconciliation purposes. It is recommended that a unique merchant reference be passed to POLi™ that can unambiguously be linked to the transaction. This may take the form of an order number, basket/trolley/cart number or any similar unique value. This field will be displayed to the user on the POLi™ pages.

Note: The character set for the merchant reference is restricted – see Section 4.2.

Australian merchants may choose to have the Merchant Reference field populated on their bank statement as an alternative to the POLi Transaction Reference Number. This is an option that if selected needs to be enabled by POLi Payments on their back end systems as opposed to an integration change. If this option is enabled then the merchant needs to ensure the data format complies with accepted bank formats (up to 18 digits alpha-numeric).

New Zealand merchants may choose to provide payee details that will assist in bank account reconciliation. The payee particulars, payee code and payee reference details may then be provided in the merchant reference. Those details will then be inserted into the corresponding payee fields on the merchant's bank statement. If the payee details are provided, they must be passed in the merchantRef field in the following format:

<payee particulars>|<payee code>|<payee reference>|<other merchant reference>

The first three fields must be provided and if this is not the case then the existing process will be assumed, resulting in NO population on the merchants bank statement. Where no value is available the merchant is to provide that placeholder with nothing in it.

If a field contains more than the allowed number of characters it will be truncated. The allowable number of characters for New Zealand is 12 characters per field, inclusive of spaces.

The MerchantReferenceFormat field is passed in the Initiate Transaction Request to allow the merchant to select the reconciliation formatting option (1 through 4).

The following MerchantReferenceFormat formats are available:

MerchantReferenceFormat	Description
1	Free format i.e. a one to one mapping of the merchant supplied reconciliation data to the Pay Anyone fields
2	The POLi ID to be mapped to the Payee Particulars field and the merchant supplied data (if any) to be mapped to the Payee Code and Payee Reference field.
3	The POLi ID to be mapped to the Payee Code field and the merchant supplied data (if any) to be mapped to the Payee Particulars and Payee Reference field.

4	The POLi ID to be mapped to the Payee Reference field and the merchant supplied data (if any) to be mapped to the Payee Particulars and Payee Code,

If the MerchantReferenceFormat is not passed in the InitiateTransaction request or a value other than 1 to 4, then the payee fields will not be populated using the MerchantRef field data. Any data in the MerchantRef field will then be displayed in the POLi pages as received.

2.1.3 Other Merchant Reference Field

The <other merchant reference> field is displayed on the Bank Selection page and POLi Receipt (if enabled), and in transaction reports – it is not populated in any bank fields.

2.1.4 Merchant Data

Merchant data is an optional field that is used to return transaction specific information to the merchant.

Examples of common merchant data include:

- Encrypted session related information that allows a merchant to "re-instate" a customer's session after the POLi™ payment has completed,
- The user ID of a logged in customer,
- The user name of a logged in customer,
- An alternative or more detailed Merchant reference to facilitate reconciliation or error tracking, and/or
- Purchase data such as product numbers or quantities.

There are no specific formatting requirements on the round trip data.

2.1.5 Database Design and Data Storage

It is recommended that each POLi™ transaction be recorded by the merchant in a database for reconciliation and customer support purposes. This may be part of the generic orders table or a specific table used only to store POLi™ transactions.

All information sent to POLi™ should be recorded against the entry and data returned by POLi™ when initiating a transaction should also be stored accordingly. Similarly, all data returned in the transaction enquiries should be recorded in order to further facilitate receipting, back office reconciliation, reporting and customer support.

2.1.6 Customer Support Details

First level customer service is a task managed by the merchant and/or its merchant group/VAR. During the registration phase you will need to contractually agree upon a Customer Service approach and supply contact information as required.

2.1.7 Use of Merchant URLs

When initiating a transaction, merchants have the ability to specify different URLs for the users to be redirected to when a transaction is finished. You can specify different URLs for each of the following scenarios:

- User has successfully completed a POLi™ payment,
- User has encountered an error when trying to make a POLi™ payment, and
- User has cancelled the POLi™ payment.

3 Merchant Site Requirements

3.1 Mandatory Checkout Elements

Merchants implementing POLi™ need to abide by POLi™ style guidelines. This is essential to provide all users a similar/consistent POLi™ experience. All style elements explained in this document will be examined by POLi Payments when POLi™ compliance testing is conducted at the end of the merchant implementation phase.

POLi™ should be presented on your checkout page alongside other payment options e.g. credit card. There are 4 mandatory POLi™ components on your checkout page:

1. POLi™ logo,
2. 'Internet Banking' or 'Internet Banking (Pay With POLi™)' heading for the POLi™ Payment section,
3. 'Learn more about POLi™' link (links to getpoli.com), and
4. 'Check the list of available banks' link (links to the POLi™ Supported Banks page).

3.2 Mandatory Receipt Elements

POLi Payments requires that merchants display the POLi ID, merchant reference, payment amount, bank receipt number (if available) and the date and time of the POLi™ payment on their receipt page. So this information is available when enquiring about a completed transaction.

Additionally, merchants also have the option to use the POLi Receipt page. This can be configured during the merchant setup in the POLi™ Console.

Please refer to the **Merchant Style Guide** for more information.

*** Please ensure that you comply with all mandatory site elements outlined in the Merchant Style Guide otherwise your site will not pass the Compliance Process.**

4 POLi™ Transaction Status

POLi™ transaction status records the progress of a transaction throughout the payment process. Each status represents a critical point in the transaction and each has to be reached before the customer can progress to the next and ultimately complete the transaction. Broadly speaking, there are two categories of transaction statuses:

Active: Attained during the course of a transaction before reaching a terminal status. A transaction with an active state is not considered finished.

Terminal: Attained when the transaction has reached a conclusion either successfully or unsuccessfully.

Status	Category	Description
Initiated	Active	When a transaction is started, it is immediately assigned the status Initiated.
FinancialInstitution Selected	Active	A transaction moves into the FinancialInstitutionSelected status when the user has selected the Financial Institution that they wish to use.
EULAAccepted	Active	When the user accepts the POLi™ End User License Agreement, the transaction moves into EULAAccepted
InProcess	Active	This status is reached when the customer hits the login page of the bank they have selected.
Unknown	Active	The transaction goes into Unknown status when the user clicks on the Confirm button to make their payment. The result of the payment is pending.
ReceiptUnverified	Terminal	A rare state attained when POLi™ cannot confirm whether the customer's payment has been successful or not. In this case, the funds may have been transferred but the customer will not be displayed a receipt from POLi™ or the merchant.
Completed	Terminal	A successful transaction where the funds have been transferred and POLi™ has recognised the confirmation.
Failed	Terminal	A transaction can go into Failed status when an error occurs during the course of the transaction. For example, the customer has an unsupported system, a connection dropout, a communication error or a similar unrecoverable problem.
Cancelled	Terminal	When the user cancels the transaction it will move into Cancelled status.
TimedOut	Terminal	Automatically attained when the customer has not completed the transaction before the POLi™ timeout and the transaction is not currently in Unknown.

Transactions with a Completed status typically result in a credit to the merchant's bank account. Transactions in Failed, Cancelled and TimedOut statuses typically will not result in a bank account credit. The status ReceiptUnverified indicates that there is uncertainty whether the purchasing customer's bank has processed the payment instruction, e.g. because communications were lost at a critical time. All other statuses indicate that the transaction is still ongoing.

4.1 Completed Transaction Disclaimer

Although every effort is made to ensure that POLi™ is robust and error free, POLi Payments cannot guarantee that funds from Completed POLi™ transactions will be credited to a merchant's bank account within any given time period. Delays in processing may occur, and, in extreme circumstances, transactions reported as Completed might never have a concluding credit to the merchant's account. Merchants must therefore be careful to reconcile POLi™ transactions regularly, preferably daily. Typically a Merchant will employ 3-way reconciliation, between the online sales system, the POLi™ daily (or period) transaction report and the destination bank account.

Some factors that can result in delayed or missing credits to the merchant's bank account for Completed transactions:

- **Incorrect destination bank account specified at registration**

  It is paramount that the merchant's destination bank account details are recorded accurately in the POLi™ system.

- **Inter-bank clearings**

  Intra-bank credits are often completed in "real time", but inter-bank clearing systems may take between 1-3 business days before funds appear in the merchant's destination bank account.

- **Processing delays**

  Occasionally, the normal course of intra-bank or inter-bank processing is delayed for various reasons. In this case, the merchant might notice that all payments originating from one financial institution are delayed on the same day.

- **Fraud prevention and investigation procedures**

  A bank may have complex fraud-related procedures which involve delaying third party payments under certain circumstances. (e.g. in the case of first time payments by a payee to a merchant, use of stolen internet banking logins or when there are many payments within a short period of time). In these rare cases, the transaction may be stopped and funds never credited to the merchant's account.

Further, there is a range of circumstances that POLi™ transactions can be reported as having "ReceiptUnverified" status. Although rare, this must be taken into account in the POLi™ implementation and reconciliation process. A "ReceiptUnverified" status indicates that POLi™ could not determine unequivocally that the internet banking payment had completed successfully.

The most prudent course of action for "ReceiptUnverified" transactions is to wait for bank reconciliation to confirm that payment has occurred. If there is no resulting credit within the expected time period, or if the matter must be resolved urgently, then joint investigations between the merchant and customer and their respective financial institutions can reveal whether the payment was successful, e.g. via production of internet banking receipt details by the customer or through a trace inquiry initiated by the merchant.

Merchants should carefully consider the risks taken with regard to the fulfilment or delivery of goods or services prior to reconciliation of credited funds.

5 Security

All communications to POLi™ must be made using HTTPS. In addition, authentication is also done on each communication attempt by way of the merchant code and authentication code.

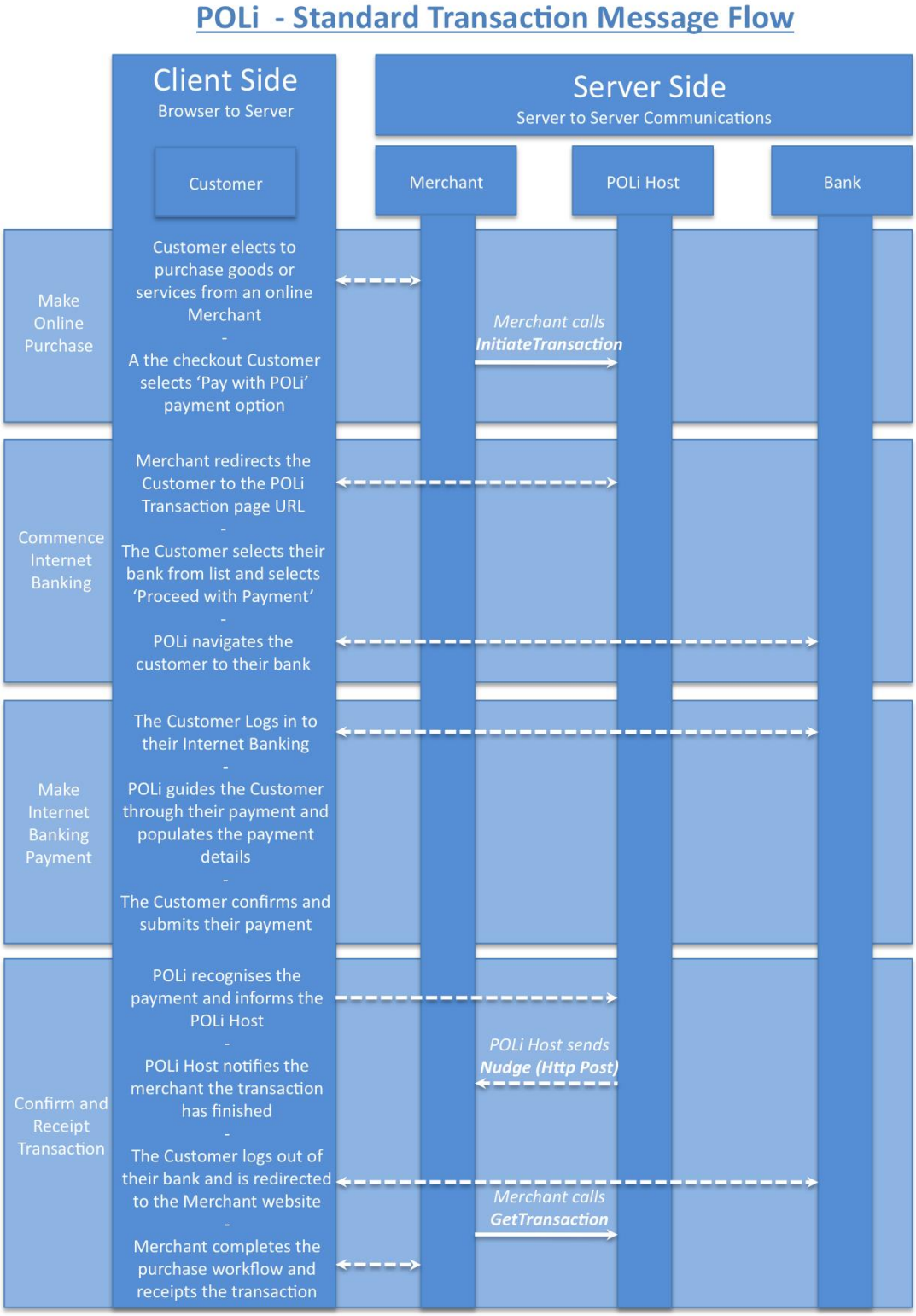
It is the responsibility of each merchant to protect their merchant code, authentication code and other confidential information. It is highly recommended that you implement security safeguards on your web site and associated infrastructure, and ensure that hosting companies or internal development teams are implementing appropriate security measures on your behalf.

Merchants should consult with relevant technical departments to ensure that the following security recommendations have been considered or implemented.

- Ensure Directory Browsing or Indexing is not enabled on your web server.
- Ensure that your shopping cart is using SSL (Secure Socket Layer) certificates.
- Merchants should always compare the amount requested with the amount paid. If these amounts differ, the corresponding transactions should be considered fraudulent and you should contact the POLi™ Contact Centre before shipping goods, etc.
- Ensure the correct file permissions are assigned to your web site directories on your web server.

6 Merchant Integration

The following diagram illustrates the message flow between the relevant servers involved in a standard POLi™ transaction.



6.1 POLi™ Merchant API

POLi™ provides a Merchant API consisting of the following web services:

- **InitiateTransaction** – used to initiate a POLi™ transaction.
- **GetTransaction** – used to acquire the status and details of a POLi™ transaction.
- **GetFinancialInstitutions** – used to acquire the list of available banks.
- **GetDailyTransactions** – used to acquire a list of transactions for a specified date.
- **GetDailyTransactionsCSV** – used to acquire a list of transactions for a specified date in a CSV format.
- **GetDetailedTransaction** – used to search for a particular transaction.

In order to access the Merchant API, you must use the secure URLs provided by the POLi™ Implementation Manager at the time of registration.

Note: Attempted communications without SSL will be rejected.

In addition, POLi™ also provides a notification service called 'POLi™ Nudge' that notifies a merchant whenever a transaction has reached a terminal status.

6.1.1 Integration Types

There are 2 different ways to access the Merchant API:

1. **SOAP Web Services**
2. **XML Web Services (REST)**

REST is a method of transmitting data over HTTP(S) without the need for additional messaging layers such as SOAP. Using REST, merchants can invoke POLi™ web services simply by posting XML data to the POLi™ REST endpoints.

There are different endpoints for the Merchant API depending on the type of integration used.

6.1.1.1 SOAP Web Services Endpoint

Australia / New Zealand:
https://merchantapi.apac.paywithpoli.com/MerchantAPIService.svc

6.1.1.2 XML Web Services (REST) Endpoint

InitiateTransaction

Australia / New Zealand:

https://merchantapi.apac.paywithpoli.com/MerchantAPIService.svc/Xml/transaction/initiate

GetTransaction

Australia / New Zealand:
https://merchantapi.apac.paywithpoli.com/MerchantAPIService.svc/Xml/transaction/query

GetFinancialInstitutions

Australia / New Zealand:
https://merchantapi.apac.paywithpoli.com/MerchantAPIService.svc/Xml/banks

GetDailyTransactions

Australia / New Zealand:
https://merchantapi.apac.paywithpoli.com/merchantapiservice.svc/Xml/transaction/daily

GetDailyTransactionsCSV

Australia / New Zealand:
https://merchantapi.apac.paywithpoli.com/merchantapiservice.svc/Xml/transaction/daily/csv

GetDetailedTransaction

Australia / New Zealand:
https://merchantapi.apac.paywithpoli.com/merchantapiservice.svc/Xml/transaction/detail

6.1.2 Web Service Character Validation

All characters are permitted in all fields, except the Merchant Reference and Authentication Code fields. To protect against attacks such as cross site scripting and SQL injection.

The Merchant Reference field is restricted to alphanumeric characters as well as the following list of acceptable characters:

- underscore (_)
- period (.)
- colon (:)
- question mark (?)
- forward slash (/)
- hyphen (-)
- pipe (|)

All others will be rejected and returned as an 8003 error – Invalid Field Characters.

The Authentication Code field is restricted to alphanumeric characters as well as the following list of acceptable characters:

- exclamation (!)
- at (@)
- hash (#)
- dollar ($)
- percentage (%)
- caret (^)
- plus (+)
-

6.2 POLi™ Payment API

POLi™ provides a Payment API consisting of the following web services:

- **GenerateURL** – used to generate a Payment URL

In order to access the Payment API, you must use the secure URLs provided by the POLi™ Implementation Manager at the time of registration

6.2.1 Integration Types

There are 2 different ways to access the Payment API:

1. **SOAP Web Services**
2. **XML Web Services (REST)**

There are different endpoints for the Payment API depending on the type of integration used.

6.2.1.1 SOAP Web Services Endpoint

Australia / New Zealand:
https://paymentapi.apac.paywithpoli.com/PaymentRouterService.svc

6.2.1.2 XML Web Services (REST) Endpoint

Australia / New Zealand:
https://paymentapi.apac.paywithpoli.com/PaymentRouterService.svc/Xml/generate

6.2.2 Web Service Character Validation

For Merchant code and Authentication code see 6.1.2

For Payment related data

- Payment Amount must be a decimal value with a precision of 2 and between the range 1.00 and 10000.00.
- Payment Reference only allows alpha numeric characters.
- Request Type allows only known values *Manual* or *Email*.
- Confirmation Email, Customer Reference allows only known values *Yes* or *No*.
- Recipient Name allows only alphabets along with the spaces, underscore, and apostrophes.
- Recipient Email must be in a valid email address format.

6.3 POLi™ Nudge

The POLi™ Nudge is an HTTP POST sent to the nominated notification URL of the merchant. The form data making up the HTTP POST contains the token (in a field marked as 'Token'), for a particular transaction which has reached a terminal state.

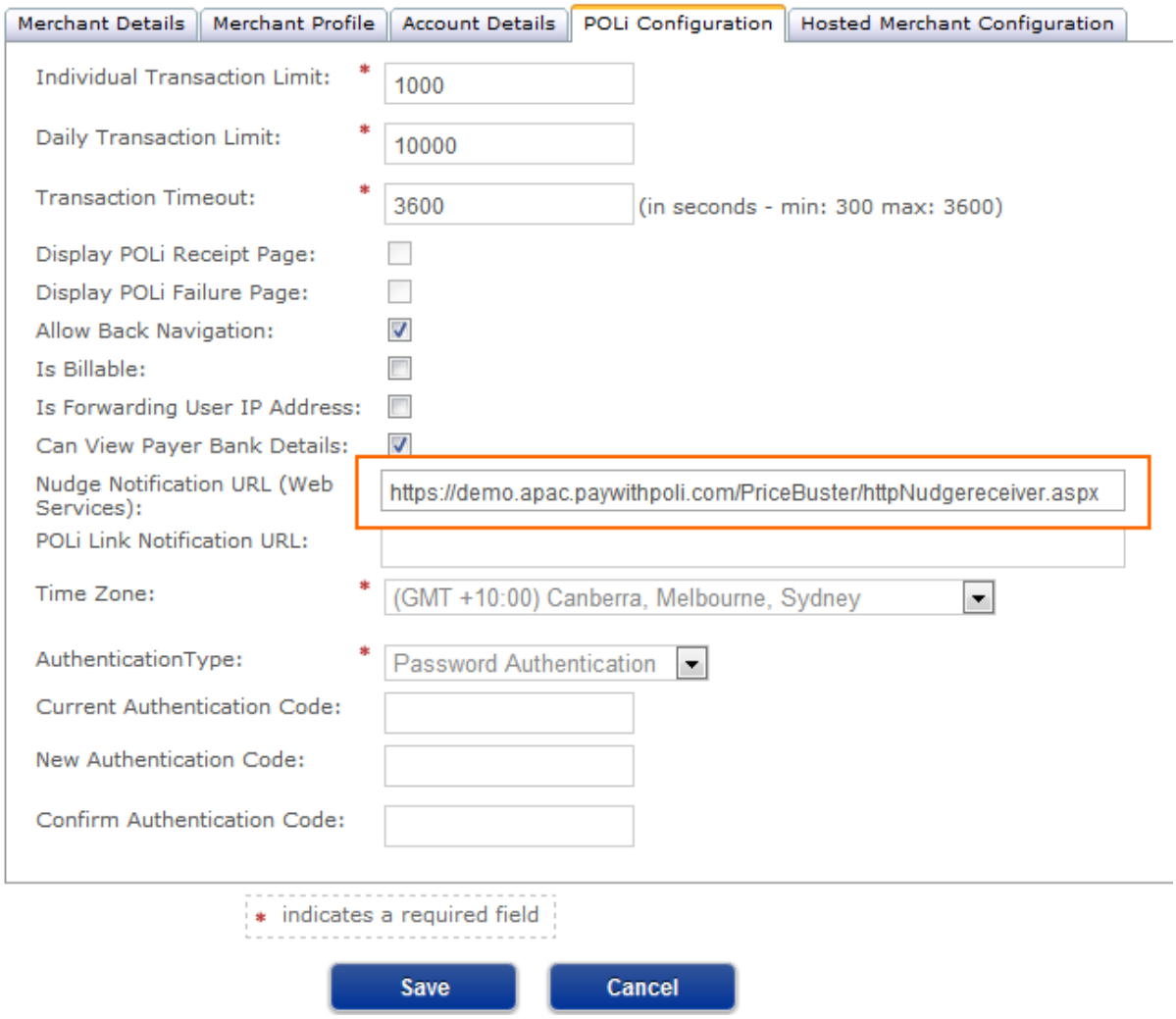
For security reasons the model used is one where the nudge contains no detailed information about the transaction, other than that it has finished. The nudge is used as a notification to the merchant that they can now enquiry about the detailed results of the transaction. **The POLi™ Nudge alone cannot be used as confirmation that a payment was successful**; it only indicates that the transaction process has now ended and finished with one of the following statuses:

Completed	A successful payment, indicating funds are to be credited to the merchant.
TimedOut	The customer did not complete the payment in the allotted time.
Failed	The customer was unable to complete the transaction.
ReceiptUnverified	POLi™ is unable to determine if the transaction was successful.
Cancelled	The transaction was cancelled by the user.

Note: The POLi™ Nudge can only be sent to the default port 80 or the secured port 443.

6.3.1 Nudge Setup

Merchants can specify that a nudge be sent to them by specifying a default notification URL in the POLi™ console, or specifying a notification URL when they initiate a transaction. If a notification URL is specified when a transaction is initiated, it will override the default notification URL specified in the POLi™ Console.



The 'Nudge Notification URL' above allows POLi™ to send a nudge for transaction in terminal state. With the above setting, upon the transaction reaching a terminal state, POLi™ will do a POST to the merchant's configured URL. The token data is URL-encoded and posted inside the Request.Form collection. The 'Nudge Notification URL' must be publicly accessible.

https://demo.apac.paywithpoli.com/PriceBuster/httpNudgereceiver.aspx

The merchant can then call GetTransaction with the token to check the status of the transaction.

6.3.2 Nudge Request

Field	Description	Data Type	Required	Allowable Values
`Token`	A time-stamped token returned by InitiateTransaction.	String	Y	Token of transaction initiated by Merchant.

6.3.3 Nudge Processing Limitations

Merchants that are unable to process the Nudge can wait until the customer is redirected to the successful or unsuccessful URL before they enquire the outcome of the transaction. The downside of this approach is that it is possible for a user to close their internet browser or have some other issue after completing the transaction in POLi™ before they are redirected back to the customer site. In this scenario, the user will never load the merchant's successful or unsuccessful URL and the merchant will not know when a transaction has completed or failed.

POLi Payments recommends that merchants adopt Nudge processing and handling where possible.

6.4 InitiateTransaction Web Service

6.4.1 Description

The InitiateTransaction web service is used to initiate a POLi™ transaction with details specified by the merchant. POLi™ will authenticate the requesting merchant and validate the data passed in the request. If successful, a POLi ID, transaction token and URL will be returned in the response.

The POLi ID is a unique 12-digits reference to a POLi™ transaction. This TRN is used in the reference field of the user bank transfer payment and should be used by the merchant to reconcile the payment. The transaction token is an encrypted form of the TRN. For security purposes, the token should be used when enquiring about the transaction in subsequent web service calls.

The URL returned by the InitiateTransaction web service is the URL that a merchant should redirect the customer to, in order to continue with their POLi™ payment. It is the URL of the POLi™ page where the customer can select the bank that they want to use to make the payment.

6.4.2 Merchant Return URLs

Merchants can specify different URLs for the users to be redirected to when a transaction is finished. There are 4 different URLs that can be specified:

- SuccessfulURL,
- UnsuccessfulURL,
- MerchantCheckoutURL, and
- MerchantHomePageURL.

6.4.2.1 SuccessfulURL

A user is redirected back to the SuccessfulURL if the user has completed a POLi™ payment successfully, and POLi™ has successfully processed the bank receipt page. POLi™ will append the transaction token as a query string parameter called 'token' in the SucessfulURL specified. For example,

- If the specified URL has no query string parameter:
  http://mywebsite.com/POLi/Receipt.aspx
  the user will be redirected to the following URL:
  http://mywebsite.com/POLi/Receipt.aspx?token=[transaction_token]

- If the specified URL has a single parameter:
  http://mywebsite.com/POLi/Receipt.aspx?param1=value1
  The user will be redirected to the following URL:

http://mywebsite.com/POLi/Receipt.aspx?param1=value1&token=[transaction_token]

- If the specified URL has multiple parameters:
  http://mywebsite.com/POLi/Receipt.aspx?param1=value1&param2=value2
  the user will be redirected to the following URL
  http://mywebsite.com/POLi/Receipt.aspx?param1=value1&param2=value2&token=[transaction_token]

6.4.2.2 UnsuccessfulURL

A user is redirected back to the UnsuccessfulURL if the user has encountered an error during a POLi™ payment and is unable to proceed. POLi™ will append the transaction token as a query string parameter called 'token' in the UnsucessfulURL specified. For example,

- If the specified URL has no query string parameter:
  http://mywebsite.com/POLi/Failure.aspx
  The user will be redirected to the following URL:
  http://mywebsite.com/POLi/Failure.aspx?token=[transaction_token]

- If the specified URL has a single parameter:
  http://mywebsite.com/POLi/Failure.aspx?param1=value1
  The user will be redirected to the following URL:
  http://mywebsite.com/POLi/Failure.aspx?param1=value1&token=[transaction_token]

- If the specified URL has multiple parameters:
  http://mywebsite.com/POLi/Failure.aspx?param1=value1&param2=value2
  the user will be redirected to the following URL
  http://mywebsite.com/POLi/Failure.aspx?param1=value1&param2=value2&token=[transaction_token]

6.4.2.3 MerchantCheckoutURL and MerchantHomePageURL

A user is redirected back to the MerchantCheckoutURL or MerchantHomePageURL if the user has the ability to proceed with the transaction but chooses not to. There are 5 different scenarios where this can occur:

- User chooses to return to the merchant on POLi™ bank selection page,
- User chooses to cancel the payment,

- User closes their Internet Banking before confirming the transaction,
- User does not have the pre-requisite installed on their machine and chooses to return to the merchant, or
- User does not have javascript enabled on their browser and chooses to return to the merchant (instead of enabling it and proceeding with the transaction).

The MerchantHomePage URL will be used if the MerchantCheckoutURL is not supplied. The MerchantHomePageURL and MerchantCheckoutURL will be used as is, that is, POLi™ will not append the transaction token to the URLs specified.

6.4.3 InitiateTransaction Request

Field	Description	Data Type	Field Length	Required	Allowable Values
`MerchantCode`	A code that uniquely identifies the Merchant requesting the transaction details	String	Max: 20	Y	The Merchant code provided by your POLi™ Implementation Manager
`AuthenticationCode`	Merchants must authenticate themselves to POLi™ via use of an authentication code (complex password)	String	Max: 500	Y	The Authentication Code must match what is set up in the POLi™ system via the POLi™ Console. The following special characters are allowed: !@#$%^+
`MerchantRef`	The merchant reference for the purchase, e.g. order number. This will be displayed to the user on the POLi™ pages.	String	Max: 100	Y	The `MerchantRef` value is restricted to alphanumeric characters and the following list of acceptable characters: • underscore (_) • period (.) • colon (:)

Field	Description	Data Type	Field Length	Required	Allowable Values
					• question mark (?) • forward slash (/) • hyphen (-)
MerchantReferenceFormat	Represents the format of reconciliation data passed in the merchant reference field and how this merchant supplied data is applied to the bank pages.	String	Max: 10	N	No data validation
MerchantData	Represents any data that the Merchant wants "round-tripped" to them during the transaction. This may be encrypted session related information that allows them to "re-instate" their Customer's session after the POLi™ payment has completed.	String	Max: 1000	N	No data validation
CurrencyCode	The currency for the transaction.	String	Max: 3	Y	There can only be one active currency for a Merchant. Any attempt to use a currency code other than that stored in the POLi™ database will result in an error. Valid currency code: AUD, NZD
CurrencyAmount	The amount of the transaction.	Decimal	Max: 20	Y	• Must be greater than zero & be fully qualified, i.e. including 2 decimal points e.g. 10.80 or 0.75 • Must not exceed system limit for currency • Must not exceed VAR limit for currency • Must not exceed Merchant single transaction limit for currency • Must not exceed Merchant daily cumulative limit for currency

Field	Description	Data Type	Field Length	Required	Allowable Values
SelectedFICode	The financial institution code selected by the user.	String	Max:20	N	The SelectedFICode must be from one of those returned by the GetFinancialInstitutions web service call. Any attempt to use a financial institution code other than that will result in an error.
MerchantDateTime	The date and time the transaction is initiated.	Datetime	Max: 30	Y	This is the time that the InitiateTransaction request is passed to POLi™. It must be within 24 hours of the POLi™'s system time or an error will result. e.g.: 2009-06-01T14:05:02
NotificationURL	The URL where the POLi™ notification (POLi™ nudge) is sent to.	String	Max: 1000	N	This must be a valid URL format.
SuccessfulURL	The URL where the user is redirected to on a successful completion of the POLi™ transaction.	String	Max: 1000	Y	This must be a valid URL format.
UnsuccessfulURL	A user will be redirected to the UnsuccessfulURL when the user has encountered an error during a POLi™ payment and is unable to proceed. Merchants need to consider what should be displayed on this page and what options should be provided to the customer.	String	Max: 1000	N	This must be a valid URL format.
MerchantCheckoutURL	The URL where the user is redirected to if the user chooses not to proceed with the POLi™ transaction. If no value is supplied, the home page URL of the merchant will be used.	String	Max: 1000	N	This must be a valid URL format.
MerchantHomePageURL	The Home Page URL of the Merchant. This is the URL where the user is redirected to if the user chooses not to proceed with the POLi™ transaction when the MerchantCheckoutURL is not supplied.	String	Max: 1000	Y	This must be a valid URL format.

Field	Description	Data Type	Field Length	Required	Allowable Values
Timeout	The number of seconds the customer has to complete the transaction. If the supplied value is 0, the default for the Merchant will be used.	Integer	Max: 4	Y	The timeout value must be an integer greater than or equal to 300 seconds (5 minutes) and less than or equal to 3600 seconds (60 minutes). If the supplied value is 0, the default for the Merchant will be used.
UserIPAddress	The IP address of the POLi™ end user, i.e. the merchant's customer.  **Note**: Merchants can specify whether they pass in the User's IP address via the POLi™ Console. If this flag is set to 'True', then this field must not be null.	String	Max: 16	Optional	• Must be a valid format, n.n.n.n format, where n is a three digit number • Must not be blacklisted on fraud list or banned country list • Need to store address to ensure that it is the same throughout the transaction

6.4.4 InitiateTransaction Response

Field	Description	Data Type	Possible Values
TransactionToken	An encrypted POLi ID	String	Unique URL-encoded string
TransactionRefNo	The POLi ID associated with the transaction.	String	A unique 12 digit reference to a POLi™ transaction.
NavigateURL	The Navigate URL is the location that the user should be redirected to, in order to proceed with the POLi™ transaction.	String	A valid URL for the POLi™ landing page appended with a query string containing the transaction token. Example: **Australia / New Zealand:** https://txn.apac.paywithpoli.com/?token=[token]
TransactionStatusCode	A status code that indicates the outcome of the transaction	String	Possible values are: • Initiated • FinancialInstitutionSelected
ErrorCode	Array of error code associated with the request.	String	See Appendix D – Error Codes

Field	Description	Data Type	Possible Values
`ErrorMessage`	Array of error message associated with the request.	String	

6.5 GetTransaction Web Service

6.5.1 Description

The GetTransaction web service is used to enquire about the details of a transaction and can be used to obtain the current status of a transaction. When supplied with a valid transaction token, GetTransaction web service returns all current details relating to a transaction. Merchants may call this web service at any time once a transaction has been initiated. The subsequent action to be taken by merchants upon receiving the GetTransaction response will depend on the transaction status returned. Centricom recommends that merchants log transaction enquiries and flag unexpected responses.

Common scenarios where GetTransaction web service is called are when a merchant is:

- displaying the SuccessfulURL page as a result of being redirected from the POLi™ transaction page,
- responding to a POLi™ nudge,
- enquiring due to approaching timeout,
- enquiring due to expired POLi™ timeout period, or
- recovering after breakdown in communications with POLi™.

6.5.2 GetTransaction Request

Field	Description	Data Type	Field Length	Required	Allowable Values
`MerchantCode`	A code that uniquely identifies the merchant requesting the transaction details	String	Max: 20	Y	The merchant code provided by your POLi™ Implementation Manager
`AuthenticationCode`	Merchants must authenticate themselves to POLi™ via use of an authentication code (complex password)	String	Max: 500	Optional	The Authentication Code must match what is set up in the POLi™ system via the POLi™ Console. The following special characters are allowed: !@#$%^+

Field	Description	Data Type	Field Length	Required	Allowable Values
TransactionToken	A time-stamped token returned by the InitiateTransaction web service.	String	50	Y	An unexpired token that corresponds to a transaction submitted by the merchant

6.5.3 GetTransaction Response

Field	Description	Data Type	Possible Values
TransactionRefNo	The POLi ID associated with the transaction.	String	A unique 12 digit reference to a POLi™ transaction.
CurrencyCode	The code of the currency used in the transaction.	String	Possible values are commensurate with ISO Standard ISO 4217.
CurrencyName	The name of the currency used in the transaction.	String	Possible values are commensurate with ISO Standard ISO 4217.
CountryCode	The code of the country where the transaction takes place.	String	Possible values are commensurate with ISO Standard ISO 3166-1.
CountryName	The name of the country where the transaction takes place.	String	Possible values are commensurate with ISO Standard ISO 3166-1.
PaymentAmount	The Amount of the transaction	Decimal	Attempted payment amount
AmountPaid	The actual amount paid for the transaction	Decimal	Actual paid amount
EstablishedDateTime	The date and time of the POLi™ server when the InitiateTransaction request was received.	Datetime	
StartDateTime	The date and time the transaction was started.	Datetime	
EndDateTime	The date and time the transaction was completed.	Datetime	
BankReceipt	The internet banking receipt number harvested from internet banking receipt page.	String	Not all banks provide bank receipt number for successful payment. This value will be null if the bank doesn't provide a receipt number.

Field	Description	Data Type	Possible Values
			If it is null, merchant can use 'TransctonStatusCode' and 'ErrorCode' to determine the payment outcome.
BankReceiptDateTime	The date and time of the bank receipt.	String	
TransactionStatusCode	A code that indicates the current status of the transaction.	String	Possible values are: • Initiated • FinancialInstitutionSelected • EULAAccepted • InProcess • Completed • Unknown • Failed • ReceiptUnverified • Cancelled • TimedOut
ErrorCode	An error code associated with the transaction, if any.	String	See Appendix D – Error Codes
ErrorMessage	A description of the error associated with the transaction, if any.	String	
FinancialInstitutionCountryCode	The country code of the financial institution the payment was made from.	String	Possible values are commensurate with ISO Standard ISO 3166-1.
FinancialInstitutionCode	The code of the financial institution the payment was made from.	String	Consult the POLi™ Console for a complete listing of supported Financial Institutions Codes and Financial Institution Names in your country
FinancialInstitutionName	The name of the financial institution the payment was made from.	String	Consult the POLi™ Console for a complete listing of supported Financial Institutions Codes and Financial Institution Names in your country
MerchantEstablishedDateTime	The merchant established date and time as passed in InitiateTransaction request.	Datetime	

Field	Description	Data Type	Possible Values
MerchantReference	The merchant reference passed in the InitiateTransaction request.	String	
MerchantDefinedData	The merchant data that was passed in the InitiateTransaction request for round trip purposes.	String	
MerchantAcctName	The merchant's account name where the funds were to be paid.	String	
MerchantAcctSortCode	The merchant's account sort code where the funds were to be paid.	String	
MerchantAcctSuffix	The merchant's account suffix where the funds were to be paid. **Note**: This is only applicable to New Zealand merchants.	String	
MerchantAcctNumber	The merchant's account number where the funds were to be paid.	String	

6.6 GetTransactionPlus Web Service

6.6.1 Description

The GetTransactionPlus web service is an extension to the GetTransaction web service. Merchants who have access to payer's bank account details may call this web service to retrieve the bank account details used to complete payment to the merchant. When supplied with a valid transaction token, GetTransactionPlus web service returns all current details relating to a transaction.

Merchants may call this web service at any time once a transaction has been initiated. The subsequent action to be taken by merchants upon receiving the GetTransactionPlus response will depend on the transaction status returned. Centricom recommends that merchants log transaction enquiries and flag unexpected responses.

Common scenarios where GetTransactionPlus web service is called are when a merchant is:

- displaying the SuccessfulURL page as a result of being redirected from the POLi transaction page,
- responding to a POLi nudge,
- enquiring due to approaching timeout,
- enquiring due to expired POLi timeout period, or
- recovering after breakdown in communications with POLi.

Participating merchants must update their web reference to MerchantAPI by refreshing the service endpoint at this location:
https://merchantapi.apac.paywithpoli.com/MerchantAPIService.svc

For enabled Merchants, GetTransactionPlus should be used as the preferred method in place of GetTransaction.

6.6.2 GetTransactionPlus Request

Field	Description	Data Type	Field Length	Required	Allowable Values
MerchantCode	A code that uniquely identifies the merchant requesting the transaction details	String	Max: 20	Y	The merchant code provided by your POLi Implementation Manager
AuthenticationCode	Merchants must authenticate themselves to POLi via use of an authentication code (complex password)	String	Max: 500	Optional	The Authentication Code must match what is set up in the POLi system via the POLi Console. The following special characters are allowed: !@#$%^&=+
TransactionToken	A time-stamped token returned by the InitiateTransaction web service.	String	50	Y	An unexpired token that corresponds to a transaction submitted by the merchant

6.6.3 GetTransactionPlus Response

Field	Description	Data Type	Possible Values
TransactionRefNo	The Transaction Reference Number associated with the transaction.	String	A unique 12 digit reference to a POLi transaction.
CurrencyCode	The code of the currency used in the transaction.	String	Possible values are commensurate with ISO Standard ISO 4217.
CurrencyName	The name of the currency used in the transaction.	String	Possible values are commensurate with ISO Standard ISO 4217.
CountryCode	The code of the country where the transaction takes place.	String	Possible values are commensurate with ISO Standard ISO 3166-1.
CountryName	The name of the country where the transaction takes place.	String	Possible values are commensurate with ISO Standard ISO 3166-1.
PaymentAmount	The Amount of the transaction	Decimal	

Field	Description	Data Type	Possible Values
AmountPaid	The actual amount paid for the transaction	Decimal	
EstablishedDateTime	The date and time of the POLi server when the InitiateTransaction request was received.	Datetime	
StartDateTime	The date and time the transaction was started.	Datetime	
EndDateTime	The date and time the transaction was completed.	Datetime	
BankReceipt	The internet banking receipt number harvested from internet banking receipt page.	String	
BankReceiptDateTime	The date and time of the bank receipt.	String	
TransactionStatusCode	A code that indicates the current status of the transaction.	String	Possible values are: • Initiated • FinancialInstitutionSelected • EULAAccepted • InProcess • Completed • Unknown • Failed • UnknownFailed • Cancelled • TimedOut
ErrorCode	An error code associated with the transaction, if any.	String	See Appendix D – Error Codes
ErrorMessage	A description of the error associated with the transaction, if any.	String	
FinancialInstitutionCountryCode	The country code of the financial institution the payment was made from.	String	Possible values are commensurate with ISO Standard ISO 3166-1.
FinancialInstitutionCode	The code of the financial institution the payment was made from.	String	Consult the POLi Console for a complete listing of supported Financial Institutions Codes and Financial Institution Names in your country
FinancialInstitutionName	The name of the financial institution the payment was made from.	String	Consult the POLi Console for a complete listing of supported

Field	Description	Data Type	Possible Values
			Financial Institutions Codes and Financial Institution Names in your country
MerchantEstablishedDateTime	The merchant established date and time as passed in InitiateTransaction request.	Datetime	
MerchantReference	The merchant reference passed in the InitiateTransaction request.	String	
MerchantDefinedData	The merchant data that was passed in the InitiateTransaction request for round trip purposes.	String	
MerchantAcctName	The merchant's account name where the funds were to be paid.	String	
MerchantAcctSortCode	The merchant's account sort code where the funds were to be paid.	String	
MerchantAcctSuffix	The merchant's account suffix where the funds were to be paid. **Note**: This is only applicable to New Zealand merchants.	String	
MerchantAcctNumber	The merchant's account number where the funds were to be paid.	String	
PayerAcctSortCode	The payer's account sort code/BSB/bank branch number used to complete POLi payment	String	
PayerAcctNumber	The payer's account number used to complete POLi payment	String	
PayerAcctSuffix	The payer's account suffix used to complete POLi Payment	String	This value is only relevant to NZ payments.

6.7 GetFinancialInstitutions Web Service

6.7.1 Description

The GetFinancialInstitutions web service is used to obtain the list of banks available to the merchant. This web service call is optional and can be used to inform the user about the list of banks that they can use with POLi™. This is useful as users can then determine if their bank is supported by POLi™ before selecting POLi™ as the payment method.

6.7.2 GetFinancialInstitutions Request

Field	Description	Data Type	Field Length	Required	Allowable Values
`MerchantCode`	A code that uniquely identifies the merchant requesting the transaction details	String	Max: 20	Y	The merchant code provided by your POLi™ Implementation Manager
`AuthenticationCode`	Merchants must authenticate themselves to POLi™ via use of an authentication code (complex password)	String	Max: 500	Optional	The Authentication Code must match what is set up in the POLi™ system via the POLi™ Console. The following special characters are allowed: !@#$%^+

6.7.3 GetFinancialInstitutions Response

Field	Description	Data Type	Possible Values
FinancialInstitutionCode	An array of Financial Institution Codes.	String	Members of the Financial Institution list associated with the merchant
FinancialInstitutionName	An array of Financial Institution Names.	String	Members of the Financial Institution list associated with the merchant
TransactionStatusCode	A status code that indicates the outcome of the transaction.	String	Not applicable for this request
ErrorCode	An array of error code associated with the request.	String	See Appendix D – Error Codes
ErrorMessage	An array of error message associated with the request.	String	

6.8 GetDailyTransactions Web Service

6.8.1 Description

The GetDailyTransactions web service is used to obtain a list of all transactions for a merchant on a particular date. This is normally used for reconciliation purposes and can be called at any time. Merchants have the option to specify the transaction status(es) that they're interested in. For example:

- If a merchant is interested only in 'Completed' transactions, they can specify 'Completed' in the TransactionStatusCode parameter when calling the GetDailyTransactions Web Service.
- If a merchant is interested only in 'Completed' and 'ReceiptUnverified' transactions, they can specify 'Completed,ReceiptUnverified' in the TransactionStatusCode parameter when calling the GetDailyTransactions Web Service.
- If a merchant is interested in all transactions, the TransactionStatusCode parameter should be left as blank when calling the GetDailyTransactions Web Service.

6.8.2 GetDailyTransactions Request

Field	Description	Data Type	Field Length	Required	Allowable Values
`MerchantCode`	A code that uniquely identifies the merchant requesting the transaction details	String	Max: 20	Y	The merchant code provided by your POLi™ Implementation Manager
`AuthenticationCode`	Merchants must authenticate themselves to POLi™ via use of an authentication code (complex password)	String	Max: 500	Y	The Authentication Code must match what is set up in the POLi™ system via the POLi™ Console. The following special characters are allowed: !@#$%^+
EstablishedDate	The date that is being requested for the daily transaction report	Datetime	N/A	Y	The Established Date must conform to this format: yyyy-mm-ddThh:mm:ss, for example: 2011-01-01T01:02:03
TransactionStatusCode	A comma separated list of transaction status codes. Send an empty string to represent all transaction statuses	String	Max: 1000	N	See description

6.8.3 GetDailyTransactions Response

Field	Description	Data Type	Possible Values
TransactionRefNo	The POLi ID associated with the transaction.	String	A unique 12 digit reference to a POLi™ transaction.
MerchantCode	Merchant Code.	String	Merchant code provided by your POLi™ Implementation Manager
CurrencyCode	The currency of the transaction.	String	Possible values are commensurate with ISO Standard ISO 4217.
CurrencyName	The currency of the transaction.	String	Possible values are commensurate with ISO Standard ISO 4217.
MerchantCommonName	The merchant common name.	String	
MerchantReference	The merchant reference passed in the InitiateTransaction request.	String	
MerchantDefinedData	The merchant data that was passed in the InitiateTransaction request for round trip purposes.	String	
PaymentAmount	The amount of the transaction.	Decimal	
AmountPaid	The actual amount paid for the transaction.	Decimal	
FinancialInstitutionCode	The code of the Financial Institution the payment was made from.	String	Consult the POLi™ Console for a complete listing of supported Financial Institutions Codes and Financial Institution Names in your country.
FinancialInstitutionName	The name of the Financial Institution the payment was made from.	String	Consult the POLi™ Console for a complete listing of supported Financial Institutions Codes and Financial Institution Names in your country.
BankReceiptNo	The internet banking receipt number harvested from internet banking receipt page.	String	Not all banks provide receipt number. This field will be null if the receipt number is unavailable.
TransactionStatusCode	A status code that indicates the outcome of the transaction.	String	Possible values are: • FinancialInstitutionSelected • EULAAccepted • InProcess

Field	Description	Data Type	Possible Values
			• Completed • Unknown • Failed • ReceiptUnverified • Cancelled • TimedOut
`TransactionStatus`	A status description that indicates the outcome of the transaction.	String	
`EstablishedDateTime`	The date and time of the POLi™ server when the InitiateTransaction request was received.	Datetime	
`EndDateTime`	The date and time the transaction was completed.	Datetime	
`ErrorCode`	An array of error code associated with the request.	String	See Appendix D – Error Codes
`ErrorMessage`	An array of error message associated with the request.	String	

6.9 GetDailyTransactionsCSV Web Service

6.9.1 Description

The GetDailyTransactionsCSV web service is used to obtain a list of all transactions for a merchant on a particular date in a CSV format. This is normally used for reconciliation purposes and can be called at any time. Merchants have the option to specify the transaction status(es) that they're interested in. For example:

- If a merchant is interested only in 'Completed' transactions, they can specify 'Completed' in the TransactionStatusCode parameter when calling the GetDailyTransactions Web Service.
- If a merchant is interested only in 'Completed' and 'ReceiptUnverified' transactions, they can specify 'Completed,ReceiptUnverified' in the TransactionStatusCode parameter when calling the GetDailyTransactions Web Service.
- If a merchant is interested in all transactions, the TransactionStatusCode parameter should be left as blank when calling the GetDailyTransactions Web Service.

6.9.2 GetDailyTransactionsCSV Request

Field	Description	Data Type	Field Length	Required	Allowable Values
`MerchantCode`	A code that uniquely identifies the merchant requesting the transaction details	String	Max: 20	Y	The merchant code provided by your POLi™ Implementation Manager
`AuthenticationCode`	Merchants must authenticate themselves to POLi™ via use of an authentication code (complex password)	String	Max: 500	Optional	The Authentication Code must match what is set up in the POLi™ system via the POLi™ Console. The following special characters are allowed: !@#$%^+
EstablishedDate	The date that is being requested for the daily transaction report	Datetime	N/A	Y	The Established Date must conform to this format: yyyy-mm-ddThh:mm:ss, for example: 2011-01-01T01:02:03

Field	Description	Data Type	Field Length	Required	Allowable Values
TransactionStatusCode	A comma separated list of transaction status codes. Send an empty string to represent all transaction statuses	String	Max: 1000	N	See description

6.9.3 GetDailyTransactionsCSV Response

Field	Description	Data Type	Possible Values
`CSVData`	Each field in the GetDailyTransactions response separated by a comma.	String	
`ErrorCode`	An array of error code associated with the request.	String	See Appendix D – Error Codes
`ErrorMessage`	An array of error message associated with the request.	String	

6.10 GetDetailedTransaction Web Service

6.10.1 Description

The GetDetailedTransaction web service is used to obtain a detailed view of a transaction. When supplied with a valid POLi ID or merchant reference, GetDetailedTransaction returns all current details relating to a payment, including transaction steps. This is normally used for support purposes and can be called at any time after the transaction is initiated.

6.10.2 GetDetailedTransaction Request

Field	Description	Data Type	Field Length	Required	Allowable Values
MerchantCode	A code that uniquely identifies the merchant requesting the transaction details	String	Max: 20	Y	The merchant code provided by your POLi™ Implementation Manager
AuthenticationCode	Merchants must authenticate themselves to POLi™ via use of an authentication code (complex password)	String	Max: 500	Optional	The Authentication Code must match what is set up in the POLi™ system via the POLi™ Console. The following special characters are allowed: !@#$%^+
MerchantReference	The merchant reference of the transaction	String	Max: 100	Optional (Required if TransactionRefNo is not supplied)	
TransactionRefNo	A time-stamped token returned in the InitiateTransaction web method.	String	Max: 50	Optional (Required if MerchantReference is not supplied)	
IncludeSteps	Whether to return the transaction steps for the transaction.	Boolean	n/a	Y	true or false (case sensitive)

6.10.3 GetDetailedTransactionResponse

DetailedTransaction Fields

Field	Description	Data Type	Possible Values
TransactionRefNo	The POLi ID associated with the transaction.	String	A unique 12 digit reference to a POLi™ transaction.
CurrencyCode	The currency of the transaction.	String	See data validation
CurrencyName	The currency of the transaction.	String	See data validation
MerchantCommonName	The merchant common name.	String	
MerchantReference	The merchant reference passed in the InitiateTransaction request.	String	
MerchantDefinedData	Data that was passed in the InitiateTransaction request for round trip purposes.	String	
PaymentAmount	The amount of the transaction	Decimal	
AmountPaid	The actual amount paid for the transaction	Decimal	
FinancialInstitutionCode	The financial institution code of the financial institution the payment was made from.	String	See data validation
FinancialInstitutionName	The financial institution name of the financial institution the payment was made from.	String	See data validation
BankReceiptNo	Internet bank receipt number harvested from internet banking receipt page.	String	Not all banks provide receipt number. This value will be null if unavailable. If this value is null, merchant can use 'TransactionStatusCode' and 'ErrorCode' to determine the payment outcome.
TransactionStatusCode	A status code that indicates the outcome of the transaction.	String	Possible values are: • Initiated • FinancialInstitutionSelected • EULAAccepted • InProcess • Completed • Unknown • Failed • ReceiptUnverified • Cancelled • TimedOut
TransactionStatus	A status description that indicates the outcome of the transaction.	String	See data validation
EstablishedDateTime	Date & time the transaction was initiated.	Datetime	

Field	Description	Data Type	Possible Values
EndDateTime	Date & time the transaction was completed.	Datetime	
UserIPAddress	IP Address of the customer as passed in initiate transaction.	String	
UserPlatform	The customer's browser version, .NET framework version and operating system version.	String	
FailureReason	Error Code + Error Description.	String	
ErrorCode	An array of error code associated with the request.	String	See Appendix D – Error Codes
ErrorMessage	An array of error message associated with the request.	String	

TransactionStepList [] Fields

Field	Description	Data Type	Possible Values
CreatedDateTime	Date and time the transaction step occurred.	DateTime	
TransactionStepTypeName	The description of the transaction step.	String	

7 Error Handling

Errors can be returned due to malformed web service requests or failure to comply with POLi™ business rules. For security concerns, the exact reason why an error is returned will not be given in the response. However, the table below provides an overview of the types of errors that may be returned to a merchant and their meaning. For a more detailed list, please refer to Appendix D.

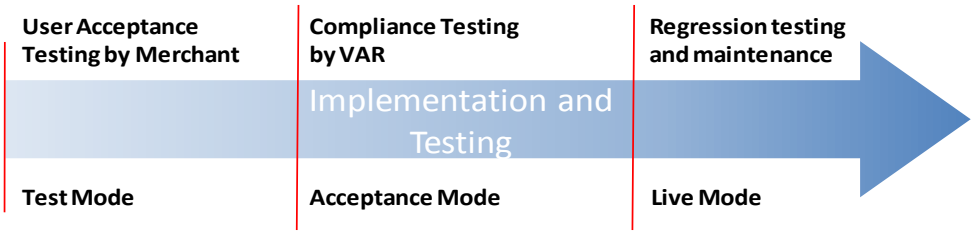
Error Code	Reason Description	Possible Reasons	Recommendation
1001-1032	Initiate Transaction Failed	• The call to initiate transaction was invalid, contained invalid, expired or illegal data or could not be verified	Check that your web services are passing the correct data in the correct format.
2001-2027 & 2029-2038	The transaction was aborted by POLi™	• Transaction data was interfered with or not as expected • Data to be used in the transaction or that was encountered through the course of the transaction was invalid	Record the Error Code in your database and redirect the customer to your failure page.
3001-3030	The transaction was aborted by the POLi™ transaction pages	• Transaction data was interfered with or not as expected by the transaction page • Data to be used in the transaction or that was encountered through the course of the transaction was invalid	Record the Error Code in your database and redirect the customer to your failure page.
5001-5002 5004 5006-5007 5010-5026	The Vector was unable to perform an action	• There was an error in the vector or the bank site had changed sufficiently to render the vector unable to perform the required action • Data to be used in the transaction or that was encountered through the course of the transaction was invalid	Record the Error Code in your database and redirect the customer to your failure page.
2028 5003 5008-5009	Invalid Certificate	• Bank Certificate SSL Validation failure • Customer could be victim of spoofing attack or could be attempting fraud	Record the Error Code in your database and redirect the customer to your failure page.
5005	Unexpected Bank Page	• The page received from the bank differs from that expected. • Customer may have been on the receipt page and future transaction enquiries may arise	Record the Error Code in your database and redirect the customer to your failure page.
6001-6005	The customer was unable to continue with the payment	• Their platform was not supported • They did not have the required pre-requisites • Javascript was disabled	Record the Error Code in your database and redirect the customer to your failure page.
8001-8007	There was a communication error and the transaction was unable to be completed	• Web service response error	Direct the customer to www.getpoli.com for more information.
10001-10009	Payment data was incorrect	• Data being submitted in the bank process was not as expected	Record the Error Code in your database and redirect the customer to your failure page.
12001-12024	Payment API errors	• Payment data submitted was not as expected.	Check that your web service call is passing valid merchant and payment data in the agreed format

8 User Acceptance Testing and Compliance Checking

8.1 Production Modes

To ensure that POLi™ integration into your site has been implemented properly and that your details have been correctly recorded in our system, Centricom implements 3 production modes. These are:

- Test Mode
- Acceptance Mode
- Live Mode



8.1.1 Test Mode

Purpose	• To test the integration of POLi™ with your website and ensure that web services are implemented properly with the correct arguments. • To perform test transactions using iBank ensuring the correct account details are retrieved and receipts are created properly with the correct information.
Bank Availability	Only iBank is available in the list of available banks.
Approval Requirements	Repeated successful compliant transactions using iBank that: • pass the correct arguments in all web service requests, • retrieve the correct transaction details from POLi™, and • display receipt with the correct data.

8.1.2 Acceptance Mode

Purpose	• To perform real transactions with real back accounts. • To reconcile against a real bank account and POLi™ Daily Transaction Reports
Bank Availability	iBank and all POLi™ supported banks for the merchant's region.
Approval Requirements	Repeated successful transactions that are fully reconciled and approved by all parties.

8.1.3 Live Mode

Purpose	Removes iBank from the list of available banks and permits POLi™ to be used as a real payment option for customers.
Bank Availability	All POLi™ supported banks for the merchant's region – No iBank.
Approval Requirements	None.

9 Support Services

The following tools are provided to support the integration, testing and production of POLi™:POLi™ Console

- Demo Merchant
- Demo Bank (iBank)
- Test Merchant Code

9.1 POLi™ Console

The POLi™ Console is a secured online portal which facilitates access to ongoing merchant support services including account administration and reporting. The POLi™ Console is primarily used for maintaining merchant account information and reporting. The main reports that can be accessed from the POLi™ Console include:

Daily Transaction Report	Used to view all reports from the specified date with a transaction end state of All, Complete or Failed. Displays the POLi ID, Merchant reference, Amount, Bank and relevant transaction information.
Summary Transaction Report	Used to view a summary of all transactions during a specified period and transaction end state. Summarises the amount paid and number of transactions.
Detailed Transaction Finder	Used to find details about individual transaction.

More information on the reports found in the POLi™ Console can be found in the POLi™ Console User Guide for Merchants.

9.2 Demo Bank (iBank)

iBank is a fictitious internet banking site which exists for merchants to simulate test transactions with no monetary value attached. It is designed to simulate the behaviour of real internet banking services.

Appendix A: SOAP Web Services (Sample Code)

The following is a sample code (in c#) that initiates a POLi™ transaction using SOAP web service. It is presented as a sample only and will not be updated or maintained.

Initiate Transaction

```
// Contact the POLi™ Application Server and get a POLi ID
InitiateTransactionRequest request = new InitiateTransactionRequest();
request.Transaction = new InitiateTransactionInput();
request.Transaction.Timeout = 1000;
request.Transaction.CurrencyCode = "AUD";
request.Transaction.MerchantDateTime = DateTime.Now;
request.Transaction.MerchantCode = "PriceBusterDVD";
request.Transaction.MerchantRef = "|PriceBuster|5000";
request.Transaction.MerchantReferenceFormat = "2";
request.Transaction.MerchantData = "MyDefinedData";
request.Transaction.CurrencyAmount = 22.50M;
request.Transaction.NotificationURL = "https://pricebusterdvd.com/NudgeReceiver.aspx";
request.Transaction.SuccessfulURL = "https://pricebusterdvd.com/Receipt.aspx?token=";
request.Transaction.UnsuccessfulURL = "https://pricebusterdvd.com/Failure.aspx?token=";
request.Transaction.MerchantCheckoutURL = "https://pricebusterdvd.com/Checkout.aspx";
request.Transaction.MerchantHomePageURL = "https://pricebusterdvd.com/Home.aspx";
request.Transaction.UserIPAddress = HttpContext.Current.Request.UserHostAddress;
request.AuthenticationCode = "$Q9h@3I51y#";

InitiateTransactionResponse response = null;
using (MerchantAPIService.MerchantAPIService proxy = new MerchantAPIService.MerchantAPIService())
{
    response = proxy.InitiateTransaction(request);
}

if (response.Errors.Length > 0)
{
    String message = String.Format(
                      "Error initiating transaction.\\nError code: {0}\\nError Message: {1}\\n\\nPlease try again.",
                      response.Errors[0].Code, response.Errors[0].Message);
    showAlert(message);
    return;
}

if (response.Transaction != null)
{
```

```
    InitiateTransactionOutput txn = response.Transaction;
    if (!String.IsNullOrEmpty(txn.NavigateURL))
    {
          // save the token and POLi ID in the merchant db with the customer details

        // Redirect the client to the POLi™ Launch Page
        // url with the token attached and encoded is returned by the web service
        Response.Redirect(txn.NavigateURL);
    }
else
{
     showAlert("Error: Could not initiate transaction.\\n\\nPlease try again.");
      return;
}
```

The following is a sample code (in c#) that enquires a POLi™ transaction using SOAP web service. It is presented as a sample only and will not be updated or maintained.

Get Transaction

```
MerchantAPIService.GetTransactionRequest request = new MerchantAPIService.GetTransactionRequest();
request.TransactionToken = token;
request.MerchantCode = "PriceBusterDVD";
request.AuthenticationCode = "$Q9h@3I51y#";

MerchantAPIService.GetTransactionResponse response = service.GetTransaction(request);

if (response.Errors != null && response.Errors.Length > 0)
{
        String message = String.Format(
             "Error getting transaction details.\\nError code: {0}\\nError Message: {1}\\n\\nPlease try again.",
             response.Errors[0].Code, response.Errors[0].Message);
        showAlert(message);
        return;
}
else if (response.Transaction != null)
{
        MerchantAPIService.Transaction transaction = response.Transaction;

       //set the fields on the receipt page with the transaction data
        lblRefNumValue.Text = transaction.TransactionRefNo;
        lblMerchantCreateTimeValue.Text = transaction.MerchantEstablishedDateTime.ToString();
        lblPaymentAmountValue.Text = String.Format("{0}{1}", GetCurrencySymbolByCountryCode(transaction.CurrencyCode),
            transaction.PaymentAmount);
        lblAmountPaidValue.Text = String.Format("{0}{1}", GetCurrencySymbolByCountryCode(transaction.CurrencyCode),
            transaction.AmountPaid);
        lblPaidFromValue.Text = transaction.FinancialInstitutionCode;
        lblPaidToAcctValue.Text = transaction.MerchantAcctName;
        lblPaidToSortCodeValue.Text = transaction.MerchantAcctSortCode;
        lblPaidToAcctNoValue.Text = transaction.MerchantAcctNumber;

        if (!String.IsNullOrEmpty(transaction.BankReceipt))
        {
            lblBankReceiptValue.Text = transaction.BankReceipt;
        }
        else
        {
```

```
            liBankReceipt.Visible = false;
        }

        if (!String.IsNullOrEmpty(transaction.BankReceiptDateTime))
        {
            lblBankReceiptedAtValue.Text = transaction.BankReceiptDateTime;
        }
        else
        {
            liBankReceiptTime.Visible = false;
        }

        //update the merchant db with the transaction details
}
```

The following is a sample code (in c#) that generates a Payment URL using SOAP web service. It is presented as a sample only and will not be updated or maintained

Generate Payment URL

```
 //PaymentAPI is the sample Namespace generated by Service Reference(or svcutil).
 //If you wish to send an email with the link, set RequestType="Email", RecipientName and RecipientEmail with valid values.

 PaymentAPI.PaymentDataRequest pdr = new PaymentAPI.PaymentDataRequest();
 pdr.MerchantCode = "PriceBusterDVD_AU";
 pdr.AuthenticationCode = "MyPassword";
 pdr.RequestType = "Manual";
 pdr.PaymentAmount = 123;
 pdr.PaymentReference = "Payment Reference";
 pdr.ConfirmationEmail = "No";
 pdr.CustomerReference = "No";

 PaymentAPI.PaymentDataResponse res = null;
 String url = string.empty;

 //PaymentAPIServiceClient is generated by Service Reference and using SoapService
endpoint
 using (PaymentAPI.PaymentAPIServiceClient svc = new PaymentAPI.PaymentAPIServiceClient("SoapService"))
 {
    res = svc.GenerateURL(pdr);
 }
```

```
if(res != null)
 {
      if(res.Errors.length>0)
      {
            String message = String.Format(
                        "Error while generating Payment URL.\\nError code: {0}\\nError Message: {1}\\n\\nPlease try
                        again.",
                              res.Errors[0].Code, res.Errors[0].Message);
        showAlert(message);
        return;

      }
      else
        url = res.PaymentURL;

  }
  else
  {
     showAlert("Error: Could not generate a Payment URL.\\n\\nPlease try again.");
     return;
  }
```

Appendix B: XML Web Services – REST (Sample Code)

This section will show an example of how a merchant can communicate with POLi™ using XML Web Services (REST).

Note: Schemas for the Merchant API Contract and Merchant API DCO can be found in Appendix C.

The sample merchant used is configured as follows:

- Merchant Code: PriceBusterDVD
- Authentication Code: MerchantPassword
- Home page URL: http://www.pricebusterdvd.com/home
- Checkout URL: http://www.pricebusterdvd.com/checkout
- Notification URL: http://www.pricebusterdvd.com/notification
- Successful URL: http://www.pricebusterdvd.com/successful
- Unsuccessful URL: http://www.pricebusterdvd.com/unsuccessful

In this scenario, a user (with IP Address '65.2.45.1') wants to pay for a purchase made on the PriceBusterDVD website using POLi™. The amount of the transaction is AUD10.00.

Initiating A New Transaction

To initiate the transaction with POLi™ using REST, the merchant needs to perform a HTTP(S) post to the designated POLi™ InitiateTransaction REST endpoint.

Troubleshooting Note: If you have trouble initiating the transaction, there are 3 key things to check.

1. Make sure you include all fields in the xml posted/sent.
2. Ensure that the field order is the same as in these examples (sometimes this matters).
3. Check dates provided are in the same format as provided in the examples below.

Initiate Transaction Request

The request content-type must be set to 'text/xml' and the following XML data included within the request body:

```
<InitiateTransactionRequest
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <AuthenticationCode>MerchantPassword</AuthenticationCode>
  <Transaction
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:CurrencyAmount>15.00</a:CurrencyAmount>
    <a:CurrencyCode>AUD</a:CurrencyCode>
    <a:MerchantCheckoutURL>http://www.pricebusterdvd.com/checkout</a:MerchantCheckoutURL>
    <a:MerchantCode>PriceBusterDVD</a:MerchantCode>
    <a:MerchantData>MerchantDataAssociatedWithTransaction</a:MerchantData>
    <a:MerchantDateTime>2008-08-18T14:01:02</a:MerchantDateTime>
    <a:MerchantHomePageURL>http://www.pricebusterdvd.com/home</a:MerchantHomePageURL>
    <a:MerchantRef>MerchantReferenceAssociateWithTransaction</a:MerchantRef>
    <a:MerchantReferenceFormat>MerchantReferenceFormat</a:MerchantReferenceFormat>
    <a:NotificationURL>http://www.pricebusterdvd.com/notification</a:NotificationURL>
    <a:SelectedFICode i:nil="true" />
    <a:SuccessfulURL>http://www.pricebusterdvd.com/successful</a:SuccessfulURL>
    <a:Timeout>1000</a:Timeout>
    <a:UnsuccessfulURL>http://www.pricebusterdvd.com/unsuccessful</a:UnsuccessfulURL>
    <a:UserIPAddress>65.2.45.1</a:UserIPAddress>
  </Transaction>
```

```
</InitiateTransactionRequest>
```

Initiate Transaction Response

Providing the request is authenticated and configured successfully, the merchant will receive the transaction token and navigateURL in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<InitiateTransactionResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
/>
  <TransactionStatusCode>Initiated</TransactionStatusCode>
  <Transaction
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">

<a:NavigateURL>https://txn.apac.paywithpoli.com/?token=%2bXo3AxIuS8T%2fukpoUCZyXw%3d%3d</a:
NavigateURL>
    <a:TransactionRefNo>996100000001</a:TransactionRefNo>
    <a:TransactionToken>+Xo3AxIuS8T/ukpoUCZyXw==</a:TransactionToken>
  </Transaction>
</InitiateTransactionResponse>
```

In the case of an error, the merchant will receive error details in the XML data as part of the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<InitiateTransactionResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:Error>
      <a:Code>1003</a:Code>
      <a:Field />
      <a:Message>POLi is unable to continue with this payment. Please contact the Merchant
for assistance.</a:Message>
    </a:Error>
  </Errors>
  <TransactionStatusCode i:nil="true" />
  <Transaction i:nil="true"
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
/>
</InitiateTransactionResponse>
```

Obtaining the Results of a POLi™ Transaction

In the example below, the merchant is enquiring on a transaction previously initiated with a transaction token of '+Xo3AxIuS8T/ukpoUCZyXw=='.

GetTransaction Request

To get the transaction details, the merchant performs a HTTP(S) post to the POLi™ GetTransaction REST endpoint with the request content-type set to 'text/xml' and the following XML data in the request body:

```
<GetTransactionRequest
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <AuthenticationCode>MerchantPassword</AuthenticationCode>
  <MerchantCode>PriceBusterDVD</MerchantCode>
  <TransactionToken>+Xo3AxIuS8T/ukpoUCZyXw==</TransactionToken>
</GetTransactionRequest>
```

GetTransaction Response

Providing the request is authenticated and configured successfully, the merchant will receive the transaction details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetTransactionResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
/>
  <TransactionStatusCode>Completed</TransactionStatusCode>
  <Transaction
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:AmountPaid>1.00</a:AmountPaid>
    <a:BankReceipt>98742364-5</a:BankReceipt>
    <a:BankReceiptDateTime>25 August 2008 14:32:32</a:BankReceiptDateTime>
    <a:CountryCode>AU</a:CountryCode>
    <a:CountryName>Australia</a:CountryName>
    <a:CurrencyCode>AUD</a:CurrencyCode>
    <a:CurrencyName>Australian Dollar</a:CurrencyName>
    <a:EndDateTime>2008-08-25T14:32:32.73</a:EndDateTime>
    <a:ErrorCode i:nil="true" />
    <a:ErrorMessage i:nil="true" />
    <a:EstablishedDateTime>2008-08-25T14:31:32.4</a:EstablishedDateTime>
    <a:FinancialInstitutionCode>iBankAU01</a:FinancialInstitutionCode>
    <a:FinancialInstitutionCountryCode>AU</a:FinancialInstitutionCountryCode>
    <a:FinancialInstitutionName>iBank AU 01</a:FinancialInstitutionName>
    <a:MerchantAcctName>Test Merchant</a:MerchantAcctName>
    <a:MerchantAcctNumber>11248877</a:MerchantAcctNumber>
    <a:MerchantAcctSortCode>123456</a:MerchantAcctSortCode>
    <a:MerchantAcctSuffix />
    <a:MerchantDefinedData>MerchantDataAssociatedWithTransaction</a:MerchantDefinedData>
    <a:MerchantEstablishedDateTime>2008-08-25T02:31:32</a:MerchantEstablishedDateTime>
    <a:MerchantReference>MerchantReferenceAssociateWithTransaction</a:MerchantReference>
    <a:PaymentAmount>1.00</a:PaymentAmount>
    <a:StartDateTime>2008-08-25T14:31:32.91</a:StartDateTime>
    <a:TransactionID>6a7f95e0-c2c6-4d45-8b46-53e63f5c218e</a:TransactionID>
    <a:TransactionRefNo>996100000001</a:TransactionRefNo>
  </Transaction>
</GetTransactionResponse>
```

If an error occurred while getting the transaction details, the merchant will get the error details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetTransactionResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
```

```
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:Error>
      <a:Code>2007</a:Code>
      <a:Field />
      <a:Message>POLi is unable to continue with this payment. Please contact the Merchant
for assistance.</a:Message>
    </a:Error>
  </Errors>
  <TransactionStatusCode>Failed</TransactionStatusCode>
  <Transaction i:nil="true"
xmlns:a=http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO />
</GetTransactionResponse>
```

Getting List of Available Financial Institutions

GetFinancialInstitutions Request

To get the list of available financial institutions, the merchant performs a HTTP(S) post to the POLi™ GetFinancialInstitutions endpoint with the request content-type set to 'text/xml' and the following XML data in the request body:

```
<GetFinancialInstitutionsRequest
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <AuthenticationCode>MerchantPassword</AuthenticationCode>
  <MerchantCode>PriceBusterDVD</MerchantCode>
</GetFinancialInstitutionsRequest>
```

GetFinancialInstitutions Response

If no error occurred while getting the financial institutions list, the merchant will get the transaction details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>

<GetFinancialInstitutionsResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">

    <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
/>

    <TransactionStatusCode i:nil="true" />

    <FinancialInstitutionList
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">

        <a:FinancialInstitution>

            <a:FinancialInstitutionCode>ANZ</a:FinancialInstitutionCode>

            <a:FinancialInstitutionName>ANZ</a:FinancialInstitutionName>

        </a:FinancialInstitution>

        <a:FinancialInstitution>

            <a:FinancialInstitutionCode>CBA</a:FinancialInstitutionCode>

            <a:FinancialInstitutionName>CBA</a:FinancialInstitutionName>

        </a:FinancialInstitution>

        <a:FinancialInstitution>
```

```
            <a:FinancialInstitutionCode>NAB</a:FinancialInstitutionCode>
            <a:FinancialInstitutionName>NAB</a:FinancialInstitutionName>
        </a:FinancialInstitution>
        <a:FinancialInstitution>
            <a:FinancialInstitutionCode>Suncorp</a:FinancialInstitutionCode>
            <a:FinancialInstitutionName>Suncorp</a:FinancialInstitutionName>
        </a:FinancialInstitution>
        <a:FinancialInstitution>
            <a:FinancialInstitutionCode>StGeorge</a:FinancialInstitutionCode>
            <a:FinancialInstitutionName>St George</a:FinancialInstitutionName>
        </a:FinancialInstitution>
        <a:FinancialInstitution>
            <a:FinancialInstitutionCode>Westpac</a:FinancialInstitutionCode>
            <a:FinancialInstitutionName>Westpac</a:FinancialInstitutionName>
        </a:FinancialInstitution>
        <a:FinancialInstitution>
            <a:FinancialInstitutionCode>iBank</a:FinancialInstitutionCode>
            <a:FinancialInstitutionName>iBankAU01</a:FinancialInstitutionName>
        </a:FinancialInstitution>
    </FinancialInstitutionList>
</GetFinancialInstitutionsResponse>
```

If an error occurred while getting the financial institutions list, the merchant will get the error details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetFinancialInstitutionsResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:Error>
      <a:Code>1005</a:Code>
      <a:Field />
      <a:Message>POLi is unable to continue with this payment. Please contact the Merchant
for assistance.</a:Message>
    </a:Error>
  </Errors>
  <TransactionStatusCode i:nil="true" />
  <FinancialInstitutionList i:nil="true"
xmlns:a=http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO />
</GetFinancialInstitutionsResponse>
```

Obtaining the Detailed Results of a POLi™ Transaction

GetDetailedTransaction Request

To get the transaction details, the merchant performs a HTTP(S) post to the POLi™ GetDetailedTransaction REST endpoint with the request content-type set to 'text/xml' and the following XML data in the request body:

```
<GetDetailedTransactionRequest
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <AuthenticationCode>MerchantPassword</AuthenticationCode>
  <MerchantCode>PriceBusterDVD</MerchantCode>
  <IncludeSteps>false</IncludeSteps>
  <MerchantReference>MerchantReference</MerchantReference>
  <TransactionRefNo>996100000002</TransactionRefNo>
</GetDetailedTransactionRequest>
```

GetDetailedTransaction Response

Providing the request is authenticated and configured successfully, the merchant will receive the transaction details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetDetailedTransactionResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
/>
  <DetailedTransaction
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:AmountPaid>0.00</a:AmountPaid>
    <a:BankReceiptNo i:nil="true" />
    <a:CurrencyCode>AUD</a:CurrencyCode>
    <a:CurrencyName>Australian Dollar</a:CurrencyName>
    <a:EndDateTime>0001-01-01T00:00:00</a:EndDateTime>
    <a:EstablishedDateTime>2008-08-22T14:07:22.023</a:EstablishedDateTime>
    <a:FailureReason i:nil="true" />
    <a:FinancialInstitutionCode>iBankAU01</a:FinancialInstitutionCode>
    <a:FinancialInstitutionName>iBank AU 01</a:FinancialInstitutionName>
    <a:MerchantCode>PriceBusterDVD_AU</a:MerchantCode>
    <a:MerchantCommonName>Pricebuster AU</a:MerchantCommonName>
    <a:MerchantDefinedData>MyDefinedData</a:MerchantDefinedData>
    <a:MerchantReference>MyRef01</a:MerchantReference>
    <a:PaymentAmount>1.00</a:PaymentAmount>
    <a:TransactionRefNo>996100000002</a:TransactionRefNo>
    <a:TransactionStatus>Eula Accepted</a:TransactionStatus>
    <a:TransactionStatusCode>EulaAccepted</a:TransactionStatusCode>
    <a:UserIPAddress>60.241.9.153</a:UserIPAddress>
    <a:UserPlatform>OS: Windows Vista, Browser: IE7.0, .NET Framework:
3.5.30428</a:UserPlatform>
  </DetailedTransaction>
  <TransactionStepList
xmlns:dco="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
>
    <a:TransactionStepsList>
      <a:CreatedDateTime>2008-08-22T14:07:22.023</a:CreatedDateTime>
      <a:TransactionStepTypeName>Status has changed to
Initiated</a:TransactionStepTypeName>
    </a:TransactionStepsList>
    <a:TransactionStepsList>
      <a:CreatedDateTime>2008-08-22T14:07:23.157</a:CreatedDateTime>
      <a:TransactionStepTypeName>Platform check has been
performed.</a:TransactionStepTypeName>
    </a:TransactionStepsList>
    <a:TransactionStepsList>
```

```
      <a:CreatedDateTime>2008-08-22T14:07:23.247</a:CreatedDateTime>
      <a:TransactionStepTypeName>The EULA has been accepted by the user through
cookie.</a:TransactionStepTypeName>
    </a:TransactionStepsList>
    <a:TransactionStepsList>
      <a:CreatedDateTime>2008-08-22T14:07:28.743</a:CreatedDateTime>
      <a:TransactionStepTypeName>Status has changed to
FinancialInstitutionSelected</a:TransactionStepTypeName>
    </a:TransactionStepsList>
    <a:TransactionStepsList>
      <a:CreatedDateTime>2008-08-22T14:07:28.793</a:CreatedDateTime>
      <a:TransactionStepTypeName>Status has changed to
EulaAccepted</a:TransactionStepTypeName>
    </a:TransactionStepsList>
    <a:TransactionStepsList>
      <a:CreatedDateTime>2008-08-22T14:07:28.793</a:CreatedDateTime>
      <a:TransactionStepTypeName>EULA</a:TransactionStepTypeName>
    </a:TransactionStepsList>
    <a:TransactionStepsList>
      <a:CreatedDateTime>2008-08-22T14:07:28.823</a:CreatedDateTime>
      <a:TransactionStepTypeName>The EULA has been accepted by the
user.</a:TransactionStepTypeName>
    </a:TransactionStepsList>
  </TransactionStepList>
</GetDetailedTransactionResponse>
```

If an error occurred while getting the transaction details, the merchant will get the error details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetDetailedTransactionResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:dco="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
>
    <a:Error>
      <a:Code>2007</a:Code>
      <a:Field />
      <a:Message>POLi is unable to continue with this payment. Please contact the Merchant
for assistance.</a:Message>
    </a:Error>
  </Errors>
  <TransactionStatusCode>Failed</TransactionStatusCode>
  <Transaction i:nil="true"
xmlns:a=http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO />
</GetDetailedTransactionResponse>
```

Obtaining a Daily List of POLi™ Transactions

GetDailyTransactions Request

To get the transaction details, the merchant performs a HTTP(S) post to the POLi™ GetDailyTransactions REST endpoint with the request content-type set to 'text/xml' and the following XML data in the request body:

```
<GetDailyTransactionsRequest
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
       <AuthenticationCode>MerchantPassword</AuthenticationCode>
       <MerchantCode>PriceBusterDVD</MerchantCode>
       <EstablishedDate>EstablishedDate</EstablishedDate>
       <TransactionStatusCode>Failed</TransactionStatusCode>
```

```
</GetDailyTransactionsRequest>
```

GetDailyTransactions Response

Providing the request is authenticated and configured successfully, the merchant will receive the transaction list in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetDailyTransactionsResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
/>
  <DailyTransactionList
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:DailyTransaction>
      <a:AmountPaid>0.00</a:AmountPaid>
      <a:BankReceiptNo i:nil="true" />
      <a:CurrencyCode>AUD</a:CurrencyCode>
      <a:CurrencyName>Australian Dollar</a:CurrencyName>
      <a:EndDateTime>2009-04-01T15:58:36.277</a:EndDateTime>
      <a:EstablishedDateTime>2009-04-01T15:58:35.62</a:EstablishedDateTime>
      <a:FinancialInstitutionCode>n/a</a:FinancialInstitutionCode>
      <a:FinancialInstitutionName>No Bank Selected</a:FinancialInstitutionName>
      <a:MerchantCode>PriceBusterDVD_AU</a:MerchantCode>
      <a:MerchantCommonName>Pricebuster AU</a:MerchantCommonName>
      <a:MerchantDefinedData>MyDefinedData</a:MerchantDefinedData>
      <a:MerchantReference>MyRef01</a:MerchantReference>
      <a:PaymentAmount>1.06</a:PaymentAmount>
      <a:TransactionRefNo>996100000006</a:TransactionRefNo>
      <a:TransactionStatus>Failed</a:TransactionStatus>
      <a:TransactionStatusCode>Failed</a:TransactionStatusCode>
    </a:DailyTransaction>
    <a:DailyTransaction>
      <a:AmountPaid>0.00</a:AmountPaid>
      <a:BankReceiptNo i:nil="true" />
      <a:CurrencyCode>AUD</a:CurrencyCode>
      <a:CurrencyName>Australian Dollar</a:CurrencyName>
      <a:EndDateTime>2009-04-01T15:58:42.667</a:EndDateTime>
      <a:EstablishedDateTime>2009-04-01T15:58:42.45</a:EstablishedDateTime>
      <a:FinancialInstitutionCode>n/a</a:FinancialInstitutionCode>
      <a:FinancialInstitutionName>No Bank Selected</a:FinancialInstitutionName>
      <a:MerchantCode>PriceBusterDVD_AU</a:MerchantCode>
      <a:MerchantCommonName>Pricebuster AU</a:MerchantCommonName>
      <a:MerchantDefinedData>MyDefinedData</a:MerchantDefinedData>
      <a:MerchantReference>MyRef01</a:MerchantReference>
      <a:PaymentAmount>1.06</a:PaymentAmount>
      <a:TransactionRefNo>996100000007</a:TransactionRefNo>
      <a:TransactionStatus>Failed</a:TransactionStatus>
      <a:TransactionStatusCode>Failed</a:TransactionStatusCode>
    </a:DailyTransaction>
  </DailyTransactionList>
</GetDailyTransactionsResponse>
```

If an error occurred while getting the transaction details, the merchant will get the error details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetDailyTransactionsResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
```

```
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:Error>
      <a:Code>2007</a:Code>
      <a:Field />
      <a:Message>POLi is unable to continue with this payment. Please contact the Merchant
for assistance.</a:Message>
    </a:Error>
  </Errors>
  <TransactionStatusCode>Failed</TransactionStatusCode>
  <Transaction i:nil="true"
xmlns:a=http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO />
</GetDailyTransactionsResponse>
```

Obtaining a Daily List of POLi™ Transactions in CSV format

GetDailyTransactionsCSV Request

To get the transaction details, the merchant performs a HTTP(S) post to the POLi™ GetDailyTransactionsCSV REST endpoint with the request content-type set to 'text/xml' and the following XML data in the request body:

```
<GetDailyTransactionsCSVRequest
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <AuthenticationCode>MerchantPassword</AuthenticationCode>
        <MerchantCode>PriceBusterDVD</MerchantCode>
        <EstablishedDate>2009-04-02T09:01:02</EstablishedDate>
        <TransactionStatusCode>Failed</TransactionStatusCode>
</GetDailyTransactionsCSVRequest>
```

GetDailyTransactionsCSV Response

Providing the request is authenticated and configured successfully, the merchant will receive the transaction list in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<GetDailyTransactionsCSVResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
/>
  <CSVData>
AmountPaid,BankReceiptNo,CurrencyCode,CurrencyName,EndDateTime,EstablishedDateTime,Financia
lInstitutionCode,FinancialInstitutionName,MerchantCode,MerchantCommonName,MerchantDefinedDa
ta1,MerchantDefinedData2,MerchantDefinedData3,MerchantDefinedData4,MerchantDefinedData5,Mer
chantDefinedData6,MerchantDefinedData7,MerchantDefinedData8,MerchantDefinedData9,MerchantDe
finedData10,MerchantReference,PaymentAmount,TransactionRefNo,TransactionStatus,TransactionS
tatusCode
    1.36,98742364-1091,AUD,Australian Dollar,2009-04-02T09:16:22.023,2009-04-
02T09:14:48.723,iBankAU01,iBank AU 01,PriceBusterDVD_AU,Pricebuster
AU,MyDefinedData,,,,,,,,,MyRef01,1.36,996100122532,Completed,Completed
  </CSVData>
</GetDailyTransactionsCSVResponse>
```

If an error occurred while getting the transaction details, the merchant will get the error details in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<GetDailyTransactionsCSVResponse
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contract
s" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Errors
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO">
    <a:Error>
      <a:Code>2007</a:Code>
      <a:Field />
      <a:Message>POLi is unable to continue with this payment. Please contact the Merchant
for assistance.</a:Message>
    </a:Error>
  </Errors>
  <TransactionStatusCode>Failed</TransactionStatusCode>
  <Transaction i:nil="true"
xmlns:a=http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO />
</GetDailyTransactionsCSVResponse>
```

Generating a Payment URL

GenerateURL Request

To generate a Payment URL, the merchant performs a HTTP(S) post to the POLi™ PaymentAPI REST URL with the request content-type set to 'text/xml' and the following XML data in the request body.

Manual request (RequestType = 'Manual')

```
<?xml version="1.0" encoding="utf-8"?>
<PaymentDataRequest xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <MerchantCode>PriceBusterDVD</MerchantCode>
        <AuthenticationCode>MerchantPassword</AuthenticationCode>
        <RequestType>Manual</RequestType>
        <PaymentAmount>123.11</PaymentAmount>
        <PaymentReference>LandingPageReferenceText</PaymentReference>
        <ConfirmationEmail>No</ConfirmationEmail>
        <CustomerReference>No</CustomerReference>
        <RecipientName></RecipientName>
        <RecipientEmail></RecipientEmail>
</PaymentDataRequest>
```

Email request (RequestType = 'Email')

```
<?xml version="1.0" encoding="utf-8"?>
<PaymentDataRequest xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <MerchantCode>PriceBusterDVD</MerchantCode>
        <AuthenticationCode>MerchantPassword</AuthenticationCode>
        <RequestType>Email</RequestType>
        <PaymentAmount>123.11</PaymentAmount>
```

```
        <PaymentReference>LandingPageReferenceText</PaymentReference>
        <ConfirmationEmail>No</ConfirmationEmail>
        <CustomerReference>No</CustomerReference>
        <RecipientName>PayerName</RecipientName>
        <RecipientEmail>PayerEmailAddress</RecipientEmail>
</PaymentDataRequest>
```

GenerateURL Response

Providing the request is authenticated and configured successfully, the merchant will receive the generated short payment URL in the XML data returned in the HTTP(S) response body as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<PaymentDataResponse>
    <Errors />
    <PaymentURL>poli.to/GMEY</PaymentURL>
</PaymentDataResponse>
```

Appendix C: Merchant API Schema For XML Web Services

Merchant API Contract XML schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="MerchantAPIService"

targetNamespace="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contracts" elementFormDefault="qualified"
xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.Contracts"
xmlns:a="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:import
namespace="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
schemaLocation="MerchantAPIServiceDCO.xsd" />

  <!-- InitiateTransaction -->
  <!-- InitiateTransaction Request -->
   <xs:element name="InitiateTransactionRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="AuthenticationCode" type="xs:string" nillable="true" />
        <xs:element name="Transaction" nillable="false">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="a:CurrencyAmount" />
              <xs:element ref="a:CurrencyCode" />
              <xs:element ref="a:MerchantCheckoutURL" />
              <xs:element ref="a:MerchantCode" />
              <xs:element ref="a:MerchantData" />
              <xs:element ref="a:MerchantDateTime" />
              <xs:element ref="a:MerchantHomePageURL" />
              <xs:element ref="a:MerchantRef" />
              <xs:element ref="a:NotificationURL" />
              <xs:element ref="a:SelectedFICode" />
              <xs:element ref="a:SuccessfulURL" />
              <xs:element ref="a:Timeout" />
              <xs:element ref="a:UnsuccessfulURL" />
              <xs:element ref="a:UserIPAddress" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <!-- InitiateTransaction Response -->
  <xs:element name="InitiateTransactionResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Errors" nillable="true">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="a:Error" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="TransactionStatusCode" type="xs:string" nillable="true"/>
        <xs:element name="Transaction" nillable="true">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="a:NavigateURL" />
              <xs:element ref="a:TransactionRefNo" />
              <xs:element ref="a:TransactionToken" />
            </xs:sequence>
```

```
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<!-- GetTransaction -->
<!-- GetTransaction Request -->
<xs:element name="GetTransactionRequest">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="AuthenticationCode" type="xs:string" nillable="true" />
      <xs:element name="MerchantCode" type="xs:string" />
      <xs:element name="TransactionToken" type="xs:string" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<!-- GetTransaction Response -->
<xs:element name="GetTransactionResponse">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Errors" nillable="true">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="a:Error" minOccurs="0" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="TransactionStatusCode" type="xs:string" nillable="true"/>
      <xs:element name="Transaction" nillable="true">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="a:AmountPaid" />
            <xs:element ref="a:BankReceipt" />
            <xs:element ref="a:BankReceiptDateTime" />
            <xs:element ref="a:CountryCode" />
            <xs:element ref="a:CountryName" />
            <xs:element ref="a:CurrencyCode" />
            <xs:element ref="a:CurrencyName" />
            <xs:element ref="a:EndDateTime" />
            <xs:element ref="a:ErrorCode" />
            <xs:element ref="a:ErrorMessage" />
            <xs:element ref="a:EstablishedDateTime" />
            <xs:element ref="a:FinancialInstitutionCode" />
            <xs:element ref="a:FinancialInstitutionCountryCode" />
            <xs:element ref="a:FinancialInstitutionName" />
            <xs:element ref="a:MerchantAcctName" />
            <xs:element ref="a:MerchantAcctNumber" />
            <xs:element ref="a:MerchantAcctSortCode" />
            <xs:element ref="a:MerchantAcctSuffix" />
            <xs:element ref="a:MerchantDefinedData" />
            <xs:element ref="a:MerchantEstablishedDateTime" />
            <xs:element ref="a:MerchantReference" />
            <xs:element ref="a:PaymentAmount" />
            <xs:element ref="a:StartDateTime" />
            <xs:element ref="a:TransactionID" />
            <xs:element ref="a:TransactionRefNo" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<!-- GetFinancialInstitutions -->
```

```
  <!-- GetFinancialInstitutions Request -->
  <xs:element name="GetFinancialInstitutionsRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="AuthenticationCode" type="xs:string" nillable="true" />
        <xs:element name="MerchantCode" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <!-- GetFinancialInstitutions Response -->
  <xs:element name="GetFinancialInstitutionsResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Errors" nillable="true">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="a:Error" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="TransactionStatusCode" nillable="true" />
        <xs:element name="FinancialInstitutionList" nillable="true">
          <xs:complexType>
            <xs:sequence>
              <xs:element ref="a:FinancialInstitution" minOccurs="0" maxOccurs="unbounded"
/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Merchant API DCO XML Schema

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
targetNamespace="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAP
I.DCO"
           attributeFormDefault="unqualified" elementFormDefault="qualified"

xmlns="http://schemas.datacontract.org/2004/07/Centricom.POLi.Services.MerchantAPI.DCO"
           xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <!-- InitiateTransaction Request DCO -->
  <xs:element name="CurrencyAmount" type="xs:decimal" />
  <xs:element name="CurrencyCode" type="xs:string" />
  <xs:element name="MerchantCheckoutURL" type="xs:string" nillable="true"/>
  <xs:element name="MerchantCode" type="xs:string" />
  <xs:element name="MerchantData" type="xs:string" nillable="true"/>
  <xs:element name="MerchantDateTime" type="xs:dateTime" />
  <xs:element name="MerchantHomePageURL" type="xs:string" />
  <xs:element name="MerchantRef" type="xs:string" />
  <xs:element name="NotificationURL" type="xs:string" nillable="true" />
  <xs:element name="SelectedFICode" type="xs:string" nillable="true" />
  <xs:element name="SuccessfulURL" type="xs:string" />
  <xs:element name="Timeout" type="xs:int" />
  <xs:element name="UnsuccessfulURL" type="xs:string" nillable="true" />
  <xs:element name="UserIPAddress" type="xs:string" nillable="true" />

  <!-- InitiateTransaction Response DCO -->
  <xs:element name="NavigateURL" type="xs:string" />
  <xs:element name="TransactionRefNo" type="xs:string" />
```

```
  <xs:element name="TransactionToken" type="xs:string" />

  <!-- GetFinancialInstitutions Response DCO -->
  <xs:element name="FinancialInstitution">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="FinancialInstitutionCode" type="xs:string" />
        <xs:element name="FinancialInstitutionName" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <!-- GetTransaction Response DCO -->
  <xs:element name="AmountPaid" type="xs:decimal" />
  <xs:element name="BankReceipt" type="xs:string" />
  <xs:element name="BankReceiptDateTime" type="xs:string" />
  <xs:element name="CountryCode" type="xs:string" />
  <xs:element name="CountryName" type="xs:string" />
  <!--xs:element name="CurrencyCode" type="xs:string" /-->
  <xs:element name="CurrencyName" type="xs:string" />
  <xs:element name="EndDateTime" type="xs:dateTime" />
  <xs:element name="ErrorCode" type="xs:string" nillable="true" />
  <xs:element name="ErrorMessage"  type="xs:string" nillable="true" />
  <xs:element name="EstablishedDateTime" type="xs:dateTime" />
  <xs:element name="FinancialInstitutionCode" type="xs:string" />
  <xs:element name="FinancialInstitutionCountryCode" type="xs:string" />
  <xs:element name="FinancialInstitutionName" type="xs:string" />
  <xs:element name="MerchantAcctName" type="xs:string" />
  <xs:element name="MerchantAcctNumber" type="xs:string" />
  <xs:element name="MerchantAcctSortCode" type="xs:string" />
  <xs:element name="MerchantAcctSuffix" type="xs:string" nillable="true" />
  <xs:element name="MerchantDefinedData" type="xs:string" />
  <xs:element name="MerchantEstablishedDateTime" type="xs:dateTime" />
  <xs:element name="MerchantReference" type="xs:string" />
  <xs:element name="PaymentAmount" type="xs:decimal" />
  <xs:element name="StartDateTime" type="xs:dateTime" />
  <xs:element name="TransactionID" type="xs:string" />
  <!--xs:element name="TransactionRefNo" type="xs:string" /-->

  <!-- Error DCO -->
  <xs:element name="Error">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Code" type="xs:string" />
        <xs:element name="Field" type="xs:string" nillable="true"/>
        <xs:element name="Message" type="xs:string" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Appendix D: Error Codes

Error Code	Description	Condition	Recommendation
1001	Invalid Token	The token specified in the request corresponds to a POLi ID that does not exist in the database.	There may be an error in the implementation of POLi™. Check your web services.
1002	Illegal Token	The token specified in the request corresponds to a POLi ID that does not belong to the specified merchant.	There may be an error in the implementation of POLi™. Check your web services.
1003	Invalid Merchant Code	The merchant code specified in the request does not exist in the database.	There may be an error in the implementation of POLi™. Check your web services.
1004	Inactive Merchant	The merchant code specified in the request corresponds to a merchant that is inactive.	There may be an error in the implementation of POLi™. Check your web services.
1005	Merchant Not Authenticated	The merchant authentication code supplied is not correct or the authentication type has not been specified in the POLi™ system.	There may be an error in the implementation of POLi™. Check your web services.
1006	Deleted Merchant	The merchant code specified in the request corresponds to a merchant that has been deleted.	There may be an error in the implementation of POLi™. Check your web services.
1007	Invalid Currency Code	The specified currency code does not exist in the database.	There may be an error in the implementation of POLi™. Check your web services.
1008	Invalid Merchant Currency	The specified currency code does not correspond to an active currency for the specified merchant.	There may be an error in the implementation of POLi™. Check your web services.
1009	Currency System Limit Exceeded	The payment amount in the specified currency has exceeded the system limit for that currency.	Inform the customer that POLi™ applies transaction limits for security reasons and to try another payment limit. Do not specify the limit.
1010	Currency VAR Limit Exceeded	The payment amount in the specified currency has exceeded the VAR limit for that currency.	Inform the customer that POLi™ applies transaction limits for security reasons and to try another payment limit. Do not specify the limit.
1011	Currency Merchant Single Transaction Limit Exceeded	The payment amount in the specified currency has exceeded the merchant's single transaction limit for that currency.	Inform the customer that POLi™ applies transaction limits for security reasons and to try another payment limit. Do not specify the limit.
1012	Currency Merchant Daily Cumulative Limit Exceeded	The payment amount in the specified currency has exceeded the merchant's daily cumulative limit for that currency.	Inform the customer that POLi™ applies transaction limits for security reasons and to try another payment limit. Do not specify the limit.
1013	Invalid Merchant Established Date Time	The difference between the specified merchant established time and the system time is more than 24 hours.	Check your date and time settings.
1014	Invalid URL Format	The format of the specified URL is invalid.	There may be an error in the implementation of POLi™. Check your web services.

1015	Invalid timeout value	The specified timeout value is less than the system minimum timeout value.	There may be an error in the implementation of POLi™. Check your web services.
1016	The transaction has expired	The transaction being enquired upon has lapsed past the 15min enquiry window	Use the Merchant Console to attain the outcome of this transaction
1017	Blocked User IP address	The IP address of the user is blocked, restricted or otherwise from a list of known suspect IP addresses	Do not try to initiate a transaction again. **NOTE:** Strongly recommend that the user not be allowed to complete the transaction using another payment option at the Merchant's discretion.
1018	Invalid IP address format	The IP address is in an invalid format	Try again passing in the correct data
1019	Invalid IP address	The IP address is invalid	Try again passing in the correct data
1020	No merchant primary account	The merchant has not set up a primary account to be used	Set up a primary account for the merchant through the console and try initiating the transaction again after the change has been approved.
1021	Invalid Field Characters	The specified field contains invalid characters.	There may be an error in the implementation of POLi™. Check your web services.
1022	Mandatory Field	No value is supplied for a mandatory field.	There may be an error in the implementation of POLi™. Check your web services.
1023	Invalid Field Length	The specified field has an invalid length.	There may be an error in the implementation of POLi™. Check your web services.
1024	Invalid Currency Amount In Field	The specified field contains invalid currency amount.	There may be an error in the implementation of POLi™. Check your web services.
1025	Invalid Field Range	The value in the field is out of the allowable range.	There may be an error in the implementation of POLi™. Check your web services.
1026	Invalid Transaction Status	The transaction has not followed the anticipated transaction status path	**NOTE:** This error should never be returned to a Merchant. If it does then please inform Centricom.
1027	Invalid Merchant Financial Institution	The Financial Institution Code passed in is not allowed for this merchant	There may be an error in the implementation of POLi™. Check your web services.
1028	Invalid Financial Institution Code	The Financial Institution Code passed in is not valid	There may be an error in the implementation of POLi™. Check your web services.
1029	Inactive Financial Institution	The Financial Institution Code passed in is not currently active	There may be an error in the implementation of POLi™. Check your web services.
1030	Deleted Financial Institution	The Financial Institution Code passed in has been deleted	There may be an error in the implementation of POLi™. Check your web services.

1031	Invalid Financial Institution Vector	The vector for the passed in Financial Institution Code is not available or non-existent	**NOTE:** This error should never be returned to a Merchant. If it does then please inform Centricom.
1032	Invalid Transaction Status Code	The Transaction Status Code passed in is not valid	There may be an error in the implementation of POLi™. Check your web services.
2021	Invalid Payment Amount	The amount being paid is not equal to the transaction amount	The user's locale can affect the way that currency amount's are displayed on the screen. For example, a German locale may display the amount with a comma rather than a decimal point. This affects the way that POLi™ interprets the amount and may result in a failure. Setting the locale to En-AU, En-US, or En-GB will rectify this issue.
8001	Operational Error Without Trace Information	An operational error occurs but there is no trace information available.	Perform the web service again.
8002	Operational Error With Trace Information	An operational error occurs and trace information is available.	Perform the web service again.
8003	Invalid Field Characters	The specified field contains invalid characters.	There may be an error in the implementation of POLi™. Check your web services.
8004	Mandatory Field	No value is supplied for a mandatory field.	There may be an error in the implementation of POLi™. Check your web services.
8005	Invalid Field Length	The specified field has an invalid length.	There may be an error in the implementation of POLi™. Check your web services.
8006	Invalid Currency Amount In Field	The specified field contains invalid currency amount.	There may be an error in the implementation of POLi™. Check your web services.
8007	Invalid Field Range	The value in the field is out of the allowable range.	There may be an error in the implementation of POLi™. Check your web services.
11002	Unable to Send Nudge	A nudge sent to the merchant has failed.	The Nudge Notification URL may be publicly inaccessible or the destination is down or it takes too long to respond.
12001	Merchant Code Required	Merchant code is empty	Provide a non-empty merchant code
12002	Merchant Code Length	Merchant code length exceeds maximum number of characters allowed.	Provide a Merchant Code with valid length.
12003 or 1003	Invalid Merchant Code	Merchant Code doesn't exist or inactive	Provide a valid Merchant Code.

12004	Authentication Code Required	Authentication Code is empty	Provide a non-empty Authentication Code
12005	Authentication Code Length	Authentication Code length exceeds maximum number of characters allowed.	Provide an Authentication Code with valid length.
12006 or 1005	Invalid Authentication Code	Authentication Code is not valid	Provide a valid Authentication Code.
12010	Request Type is Required	Request Type is empty	Provide a non-empty Request Type
12011	Request Type Length	Request Type length exceeds maximum number of characters allowed.	Provide a Request Type with a valid length
12012	Invalid Request Type	Request Type is not valid. It must be either Manual or Email.	Provide a Request Type set to either Manual or Email.
12013	Invalid Payment Amount.	Payment Amount is empty. Either it is empty or not a decimal number or decimal precision exceeds 2 or not in the range of 1.00 and 10000.00	Provide a valid Payment Amount.
12014	Payment Reference is required	Payment reference is empty.	Provide a non-empty Payment Reference.
12015	Invalid Payment Reference	Payment reference is not valid.	Provide a Payment Reference with a valid length.
12016	Invalid Confirmation Email.	Confirmation Email is either empty or not one of the values Yes, No	Provide Confirmation Email with a valid value.
12017	Invalid Customer Reference.	Customer Reference is either empty or not one of the values Yes, No	Provide Customer Reference with a valid value.
12018	Recipient Name is required.	Recipient Name is empty.	Provide a non-empty Recipient Name.
12019	Recipient Name Length	Recipient Name length exceeds maximum number of characters allowed	Provide a Recipient Name with a valid length.
12020	Invalid Recipient Name.	Recipient Name is not valid.	Check invalid characters in the Recipient Name.
12021	Recipient Email is required.	Recipient Email is empty.	Provide a non-empty recipient email.
12022	Recipient Email Length	Recipient Email length exceeds maximum number of characters allowed.	Provide a Recipient Email with a valid length.
12023	Invalid Recipient Email.	Recipient Email is not valid.	Check Recipient Email follows email address rules.
12024	Email Delivery Failed.	Email Delivery to Payer failed.	Check payer email address is valid and active.

Appendix E: User Acceptance Testing and Compliance Checking

User Acceptance Testing

Introduction

User Acceptance testing is to be conducted when a merchant completes their implementation of POLi™, prior to it being offered as a payment option. The main purpose of acceptance testing is to identify any aspects of the implementation that do not comply with POLi™ license agreement (terms and conditions specified in the Merchant Services Agreement).

Acceptance testing is beneficial to merchants, as it may uncover usability elements that can be addressed or improved before promoting the merchant web site live and releasing it to “real” users.

Acceptance Testing Categories

POLi™ acceptance testing identifies areas that can be categorized as follows:

- **Mandatory elements** - Required elements as per the Merchant Services Agreement.
- **Highly recommended elements** - Elements that are highly recommended to offer a user-friendly and secure POLi™ experience.

Mandatory Elements

Mandatory elements listed below must be addressed before POLi™ goes live:

Item	Test Details	Compliant (Yes/No?)
	An approved POLi™ Pay Online logo appears for payment option on the merchant checkout or equivalent page. The POLi™ logo must also have the following attributes: **Alt Text:** Internet Banking "Pay with POLi™" Please refer to http://www.polipayments.com/merchant_resources for approved logos	
Check the list of available banks	A "Check the list of available banks" hyperlink appears on the checkout page and displays POLi™ bank list in a new browser window	
Find out more about POLi	A "Find out more about POLi™" hyperlink appears and displays POLiPayments site (www.polipayments.com) in a new browser window	
Banking Session executes correctly	- The correct details are inserted into the payee fields - The correct details are inserted into the payment fields When transaction is complete, Internet banking session closes and user is redirected to the Merchant confirmation/receipt page.	
Merchant Return URL – User cancels transaction	A user is redirected back to the MerchantCheckoutURL or MerchantHomePageURL After cancelling the transaction - User has the ability to proceed with the transaction but chooses not to.	
Merchant Return URL – Failure 	A user is redirected back to the UnsuccessfulURL if the user has encountered an error during a POLi™ payment and is unable to proceed. - User is able to retry or select another payment method without having to repurchase or re-enter customer details	

Item	Test Details	Compliant (Yes/No?)
Transaction attempts appear on Daily Transaction Report	Upon completing test transactions using the demo bank, check that transactions (successful or not) are appearing on the daily transaction report.	
Your Order Amount : 1.00 Merchant Reference : 0000972 Bank Receipt Number : 190406154 POLi ID : 610100004296 Payment Date : 19/04/2006 Payment Time : 14:11:34	The merchant receipt/confirmation page has the following pieces of information displayed: - Amount - Merchant Reference Number (please ensure this is consistent with what appears on transaction pages. - POLi ID - Bank receipt number (only in AU) - Payment Date - Payment Time	

Highly recommended Elements

Below are highly recommended tests you should perform on your site:

Item	Details	Compliant (Yes or No?)
Real Transaction	Perform a transaction using a “real” bank. Contact Centricom to move your account from Test Mode to Acceptance Mode. **Note**: To perform a real transaction you will require the bank login details for the bank you are testing and money will be transferred to the bank account specified in your merchant account.	

10 Troubleshooting

10.1 Internal Error or Bad Request response in response to an API call

If you have trouble initiating a transaction there are 3 key things to check.

1. Make sure you include all fields in the xml posted/sent
   ie. Don't leave out null fields
2. Ensure that the field order is the same as in these examples
   sometimes the order of the fields matters
3. Check dates provided are in the same format as provided in the examples below
   Format iso8601 `2008-08-18T14:01:02`