Oracle FLEXCUBE Direct Banking

External Payment Interface

Release 12.0.3.0.0

Part No. E52543-01

April 2014

ORACLE®

External Payment Interface

April 2014

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway
Goregaon (East)
Mumbai, Maharashtra 400 063
India
Worldwide Inquiries:
Phone: +91 22 6718 3000
Fax:+91 22 6718 3001
www.oracle.com/financialservices/

Contents

1. Preface

1.1 Intended Audience

Any interested party working on the delivery of Oracle FLEXCUBE Direct Banking may read this document. The following profile of users would find this document useful:

- Application Architects
- End to End Designers
- Business Service Detailed Designers and Developers
- Implementation Partners

Specifically, however, this document is targeted at:

- **Implementation Partners, Customization Development Teams or Vendors** providing customization, configuration and implementation services around the Oracle FLEXCUBE Direct Banking product.

1.2 Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

1.3 Access to OFSS Support

https://support.us.oracle.com

1.4 Structure

This document is organized into the following chapters:

Section 1 *Preface* gives information on the intended audience. It also lists the various chapters covered in this document.

Section 2 *Provides the information about basic concept of External Payment Interface.*

Section 3 *Provides the information regarding configuration in Oracle FLEXCUBE Direct Banking for External Payment Interface.*

Section 4 *Provides the information about Oracle FLEXCUBE Direct Banking for External Payment Interface components.*

1.5 Related Information Sources

NA

2. Introduction

EPI functionality enables the business users to make merchant payments by accessing the FCDB application through a merchant site. A business user would log into any on-line e-commerce site and selects the items/bills to be purchased / paid. The e-commerce site shall display the options through which the amount can be paid by the user. The business user can invoke FCDB application upon selecting the appropriate option on the merchant's site and pay the amount by debit to their account. User will have to login to FCDB application through their login credentials and make a payment from accounts mapped to the user id. FCDB shall send the debit request to the Host and shall receive the response from the Host. Depending on the host response, FCDB shall send the success or failure status back to the merchant online.

Business user shall only be able to make the payment to the merchant. No other menu / transaction (except logout) shall be accessible / displayed to the user.

2.1 Technical Architecture – External payment Interface

The EPI module is divided into two sections as follows:

1. Admin Maintenance
    - Merchant Maintenance
2. Business Functions
    - Fund Transfer

2.2 Merchant maintenance

- For any merchant site to make use of the EPI module of FCDB, the merchant has to be first registered in FCDB. The bank administrator is provided with below transaction for registering and administrating the merchant.
    - Create Merchant
    - Modify Merchant
    - Delete Merchant
    - View Merchant

- A merchant is identified in FCDB based on a unique merchant code. The merchant code is defined by Bank Administrator at the time of registering the merchant.
- If an external payment request is received by FCDB without proper merchant code, then it is rejected.
- Based on the merchant's relationship with the bank a CASA or GL account will be mapped against the merchant code. This account will be used as credit account when the actual payment is done.
- Merchant also needs to provide success and failure URLs at the time of registration. These URLs are used by FCDB to send the responses to merchant site after the payment is processed.

ⓘ Please note the Merchant Maintenance currently follow the old FCDB framework and hence are processed by the Transaction Bean.

2.3 External Payment interface

FCDB exposes a URL "/merchant" for processing the external payment requests. This URL is mapped to the EPIServlet which internally processes all requests received from merchant site. The requests can be of two types:

- Purchase – where actual payment is made
- Verification – where the merchant requests for the status of a payment processed earlier. The merchant provides the details such as original request number, amount and FCDB reverts with the status of the original request

These processes are explained in detail in below sections

Purchase Flow

1. Merchant hits the servlet with set of http fields.
2. doPost() method of EPIServlet is called.
3. In doPost, first the idtxn is checked. If **fldTxnId** is present and it contains PUR otherwise check the same in init param **EPI.TXNID**.
4. Secondly, Merchant code is checked from http field **fldMerchCode,** if not found then **EPI.MERCHANT.CODE** init param is checked for the merchant code. Merchant code can be configured in init params in case you are providing different URLs to merchants.
5. If Merchant Code is not found, user is redirected to error page.

6. If Merchant code is found**, MerchantCode_EPIRULE** is verified against **mstrouterrule** to check if Merchant specific rule is defined, if found then this rule is used otherwise **DEFAULT_EPIRULE** is used**.**
7. Connect is called by passing request xml with the rule found in step 6, Connect provides the response xml containing UDF pairs of the parameters configured in the rule.
8. Response from connect is added to the original request and pre-login service is called.
9. For Pre-login service, validation engine is configured with custom validator for merchant code. Merchant code is validated against mstmerchantmaster. If not found error is returned from validation engine and **redirectEOT.xsl** is loaded, which shows the error message to user, and on ok click, user is redirected local error JSP.
10. On successful validation, service is called where merchant URLs are loaded from merchant master, One EPI DTO is created with all the parameters passed from merchant and it is stored in **fileupload** table.
11. Post service call, login page is loaded. After entering the credential, when user clicks enter, EPI login service is called, which internally calls the login service and loads the EPI DTO from fileupload table.
12. User logs in into the system and he is shown transaction page with all read only fields except source account.
13. On transaction screen, merchant fields are shown that were sent in request as per configuration. This is done using merchant specific XSL. If merchant specific XSL does not exist, then default configuration is loaded. (DefaultMerchantFieldConfiguration.xsl)
14. On confirmation screen, User is redirected to **EPIRedirect.jsp** from where user is directed to merchant URLs as per configuration.

Verification Flow

1. Merchant system hits the verify URL (in case separate URL is configured, separate URL configuration is required in case you are passing txn id from init params)
2. Step 2 to 8 are same as login flow
3. EPITxnVerifyService is called where system is transaction is verified from database using reference and amount details provided.
4. XML content generator is configured for verify. In the XSL, merchant specific xml can be posted back.

3. Configurations

Below are the configurations required for setting up EPI module in FCDB

3.1 web.xml configuration

Web.xml needs to be configured for mapping EPI servlet with merchant urls and passing required init parameters. Following parameters needs to be configured depending upon the requirement.

Init param	Values	Purpose	Mandatory
EPI.TXNID	**PUR/VRF**	Tells weather txn is for purchase or to verify the purchase.	Yes if field **fldTxnId** is not coming in the http request sent by merchant.
EPI.MERCHANT.CODE	**Alphanumeric**	Code of the merchant as maintained in the merchant table	Yes if field **fldMerchCode** is not coming in http request sent by merchant
EPI.VRF.XML.FLG	**Y/N**	Tells weather Verify http request contains details in xml form in some http field	Yes for verify servlet mapping.
EPI.VRF.XML.PARAMNAME	**e.g fldxml**	Contains xml where all details are stored for verify request.	Yes if EPI.VRF.XML.FLG is Y.

Attached below is the sample web.xml file. This sample file has a URL "/merchant" configured for merchants. It also has a specific URL "/rediffpur" configured for Rediff. Similarly it has one URL "/merchantverify" for verification and a separate URL "/rediffverify" for Rediff.

web.xml

3.2 Connect rule configuration

- Connect rule is required to transform the http fields passed by merchant into the format required by the FCDB application. It basically transforms the field names passed by merchant into the field names expected by the application.

- One default rule is created for all merchants. In case merchant field configuration is different from the existing default configuration, then new rule can be created for the specific merchant. Name of the rule created for specific merchant must be **<MERCHANTCODE>_EPIRULE.**
- If merchant specific rule is not found, then **DEFAULT_EPIRULE** is applied.

4. EPI components

Important web pages, services and Servlets added in FCDB framework for EPI purpose are as below:

Class or webpage	Purpose
EPIServlet.java	Main servlet to which handles all merchant requests
EPILoginService.java	Service handles pre-login and login.
EPITxnVerifyService.java	Service handles the verify requests.
EPIConstants.java	Interface for defining EPI specific constants
EPIRedirect.jsp	For redirecting user to merchant URLs
EPILogin.html	Login html
EPIRedirectError.jsp	Used for redirecting user in case validation failure for pre-login service.
EPITxnFrame.htm	For loading transaction frame
EPI_Frame.htm	EPI transaction top frame
Loginprepare.xsl	Login page loaded in this after pre-login service call(RRLGN59)
loginepisuccess.xsl	Payment screen is loaded in this XSL using EPI_Frame.htm and EPITxnFrame.htm.(RRLGN60)
DefaultMerchantFieldConfiguration.xsl	For defining merchant fields that needs to be shown on payment page. Also hidden fields can be defined in this for copying them back to merchant. Name of the field must start with epi.

4.1 Connect RULE maintenance

For configuring EPI Rule, five XMLs are needs to be uploaded using the **RuleMaint.java.** One is main rule xml and others are fragments. For configuring different rule for some merchant, only main rule needs to be uploaded again with changes in fields as per merchant requirement.

All the steps listed in this section will be taken care as part of Day 0 setup for DEFAULT_EPIRULE. If any additional rule is to be configured for specific merchant, these steps should be followed by the implementer during implementation time.

Component	Table affected	Command	XML File
DEFAULT_EPIRULE	mstrouterrule	U DEFAULT_EPIRULE 1 Y DEFAULT_EPIRULE com.iflex.fcat.connect.router.EPIRouterHandler D:\DEFAUL_RULE.XML "	 DEFAULT_EPIRULE.xml

EPIFRAG (Fragment)	mstrouterrulefragment	FU DEFAULT_EPIRULE 1 Y DEFAULT_EPIRULE com.iflex.fcat.connect.router.EPIRouter Handler D:\EPIFRAG.XML"	 EPIFRAG.XML
INCRFRAG(Fragment)	mstrouterrulefragment	FU DEFAULT_EPIRULE 1 Y DEFAULT_EPIRULE com.iflex.fcat.connect.router.EPIRouter Handler “D:\ INCRFRAG.XML"	 INCRFRAG.XML
TEMPLCOPYFRG(Fragment)	mstrouterrulefragment	FU DEFAULT_EPIRULE 1 Y DEFAULT_EPIRULE com.iflex.fcat.connect.router.EPIRouter Handler “D:\TEMPLCOPYFRG.XML"	 TEMPLCOPYFRG.X
TEMPLINITFRG(Fragment)	mstrouterrulefragment	FU DEFAULT_EPIRULE 1 Y DEFAULT_EPIRULE com.iflex.fcat.connect.router.EPIRouter Handler “D:\TEMPLINITFRG.XML"	 TEMPLINITFRG.X

5. APPENDIX