Account Opening Interface Specification Low Level Design Document

**EIDIKOSYSTEMS INTEGRATORS**



Version: 1.0

Release Date:11-08-2020

|  |  |  |
| --- | --- | --- |
| Prepared By | Reviewed By | Approved By |
| EIDIKO |  |  |

Document Classification: CONFIDENTIAL

Contents

[1 Introduction 5](#_Toc50118045)

[1.1 General Description 5](#_Toc50118046)

[2 Purpose 5](#_Toc50118047)

[3 Scope 5](#_Toc50118048)

[3.1 In Scope 5](#_Toc50118049)

[4 Channels Involved 5](#_Toc50118050)

[5 Interface Dependencies 6](#_Toc50118051)

[5.1 External Dependencies 6](#_Toc50118052)

[5.2 Internal Dependencies 6](#_Toc50118053)

[6 Business Process Summary 7](#_Toc50118054)

[6.1 Process Overview 7](#_Toc50118055)

[Figure: 1 Process flow Approach for Account Opening. 7](#_Toc50118056)

[6.1.1 Account Opening Steps 7](#_Toc50118057)

[7 IIB Implementation Process Flow 8](#_Toc50118058)

[7.1 Account Opening Successful Request Response Flow 8](#_Toc50118059)

[7.1.1 Process Diagram 8](#_Toc50118060)

[7.1.2 Process Flow Steps 8](#_Toc50118061)

[7.2 Retry Process workflow 9](#_Toc50118062)

[7.2.1 Process Flow Diagram 9](#_Toc50118063)

[7.2.2 Process Flow steps 10](#_Toc50118064)

[7.3 Raw\_Audit Process Flow: 10](#_Toc50118065)

[7.3.1 Process Flow Diagram: 10](#_Toc50118066)

[7.3.2 Process Flow Steps: 10](#_Toc50118067)

[7.4 HTTP Router Process Flow 11](#_Toc50118068)

[7.4.1 Incoming Router Process Flow Diagram: 11](#_Toc50118069)

[7.4.2 Process Flow Steps: 11](#_Toc50118070)

[7.4.3 STEWARDBANK ESB TRANSACTION DETAILS 12](#_Toc50118071)

[7.4.4 Outgoing Router Process Flow Diagram 13](#_Toc50118072)

[7.4.5 Process Flow Steps: 13](#_Toc50118073)

[7.5 Error\_Audit Process Flow 14](#_Toc50118074)

[7.5.1 Process Flow Diagram: 14](#_Toc50118075)

[7.5.2 Process Flow Steps: 15](#_Toc50118076)

[8 Interface Definitions 15](#_Toc50118077)

[8.1 Request Message Definition 15](#_Toc50118078)

[8.1.1 Channel Request Message Structure /Schema 15](#_Toc50118079)

[8.1.2 Channel Request Message Details 15](#_Toc50118080)

[8.1.3 Channel Sample Source Messages 16](#_Toc50118081)

[8.1.4 T24 Request Message Structure /Schema 16](#_Toc50118082)

[8.1.5 T24 Response Message Details 16](#_Toc50118083)

[8.1.6 T24 Sample Source Messages 17](#_Toc50118084)

[8.2 Response Message Definition 19](#_Toc50118085)

[8.2.1 Channel Response Message Structure/Schema 19](#_Toc50118086)

[8.2.2 Channel Response Message Details 19](#_Toc50118087)

[8.2.3 Channel Sample Response Messages 20](#_Toc50118088)

[8.2.4 Channel Error Response Message 22](#_Toc50118089)

[8.2.5 Data Base Error Response Codes 23](#_Toc50118090)

[9 Logging Mechanism 24](#_Toc50118091)

[9.1 Insert into RAW\_AUDIT\_TABLE 24](#_Toc50118092)

[9.2 Insert into ERROR\_AUDIT\_TABLE 25](#_Toc50118093)

[9.3 File Based Logging 26](#_Toc50118094)

[9.3.1 Log4j 26](#_Toc50118095)

**Approval**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Project Role | Approver | Company | E-mail | Sign-Off Date |
|  |  | Eidiko |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Description | Version | Author |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

## General Description

This Interface Design document outlines the integration requirements for the Account Opening ***(Request and Response)***interface used as banking operations. It summarizes the business processes, which use this interface. It also covers error handling and exception scenarios.

# Purpose

The purpose of this document is to capture events that trigger the interface, main steps within the interface and the integration architecture. This document is intended for use by the developers of the applications identified, the integration development team, and by the test organizations responsible for the testing of these applications.

# Scope

## In Scope

This document focuseson outlining the interface design for the Account Opening ***(Request and Response)***interface. Central to this document are the following:

* Overview of the business process that drives the need for the interface
* Proposed integration approach
* Trigger events and business dependencies on this interface
* Sequence /Flow Chart diagram of the interface
* Error handling and exception scenarios
* Validation and backup requirements

# Channels Involved

The following table lists Currency Converter channels

|  |  |
| --- | --- |
| **Item #** | **Channel Name** |
| 1 | USSD |

# Interface Dependencies

## External Dependencies

The following table lists interface specific External requirements.

|  |  |
| --- | --- |
| **Item #** | **External Requirements** |
| 1 | Steward Database – (SQLDB) |
| 2 | Core Banking System(Temenos-T24) |
| 3 | Postilion System |

## Internal Dependencies

The following table lists interface specific internal requirements.

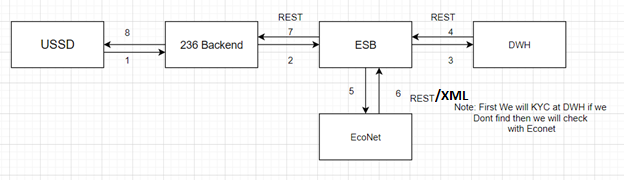
|  |  |
| --- | --- |
| **Item #** | **Internal Requirements** |
| 1 | HTTP Router Interface (Gateway) - StewardBankHttpRoutersApp |
| 2 | HTTP Retry Interface (Subflow) - StewardBankHttpRetryCall |
| 3 | DB Logging App (Audit Logging) |
| 4 | StewardBankCommonEsql (Shared Library) |

# Business Process Summary

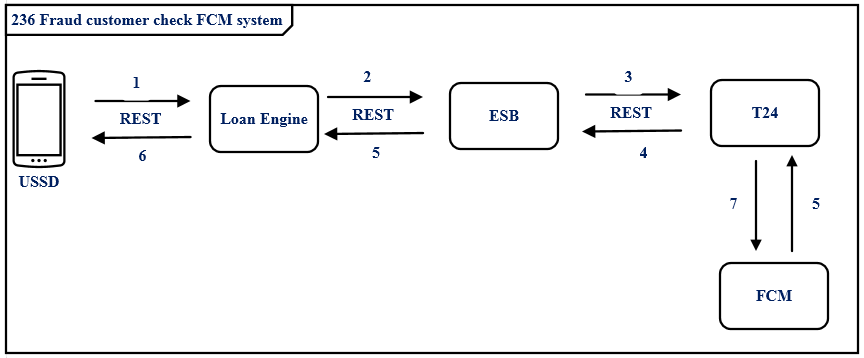
## Process Overview

1. **Account Opening (Low KYC Customer):** The **Account Opening** process primarily involves to 3 steps
2. Confirming the KYC of the customer
3. Fraud Customer Check with FCM System
4. Updating the Account Details in ZSS/Postilion
5. **KYC Customer:** This flow is to check   
   whether the customer has KYC in DWH or ECONET.

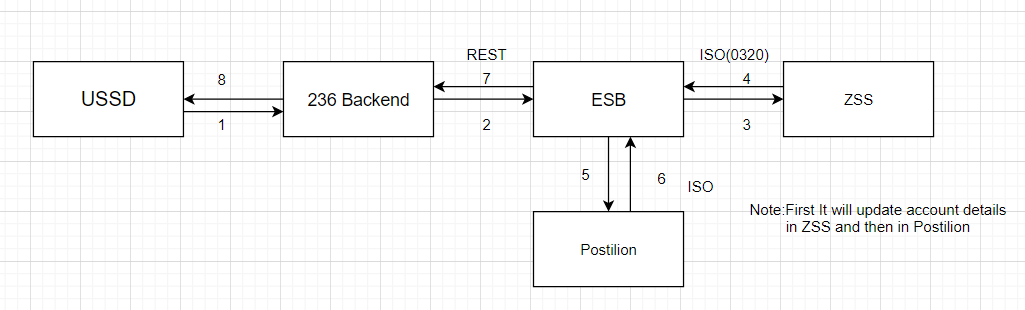
If it can find KYC in the DWH then no need of calling ECONET



1. **FCM:** Once we send the KYC found in either ECONET or DWH then Customer will confirm whether that KYC belongs to him or not, If he confirm then FCM call will takes place to check whether customer is fraud or not.



1. **Update ZSS and Postilion:** If the Customer is not fraud then need to update the Account Details assigned for the customer in ZSS and Postilion.



### Figure: 1 Process flow Approach for Account Opening.

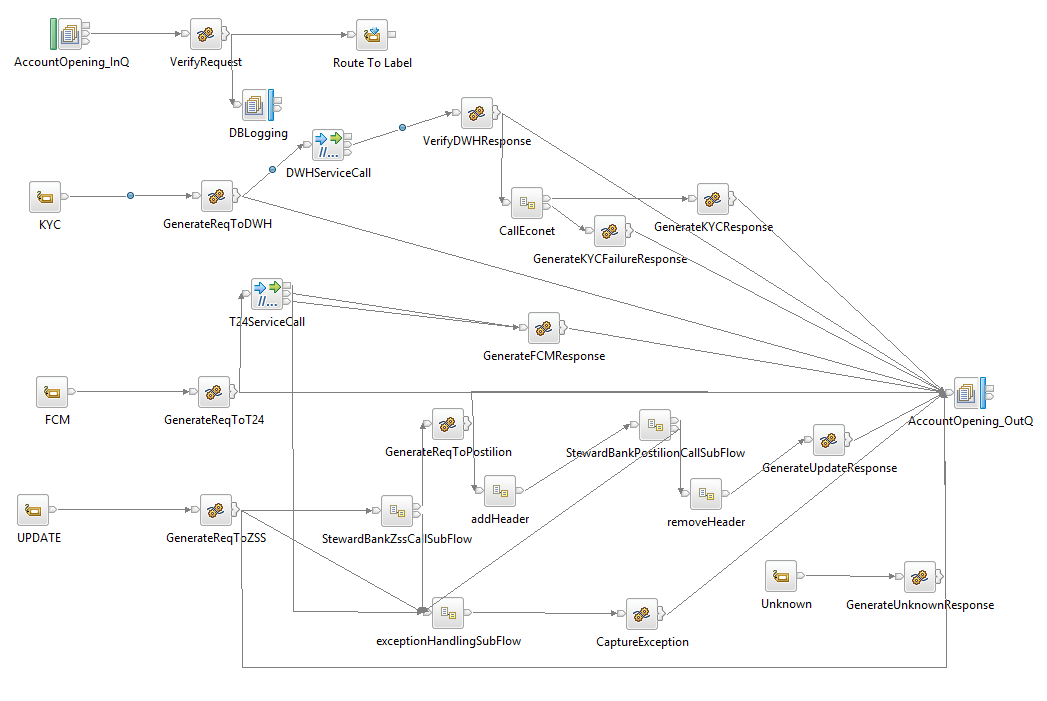
### Account Opening Steps

|  |  |
| --- | --- |
| **S. No** | **Activities** |
| 1 | ESB receives the request from channel (236 Backend). |
| 2 | Request channel is the http request accepts the request from the channel |
| 3 | ESB frames request for checking KYC with data warehouse (DWH) through a web service call. If there is no such customer, then ESB frames a request to check KYC with Eco net and either it is success or failure the response will be send back channel. |
| 4 | ESB frames a request for Fraud Customer Check (FCM) to verify whether the customer is having any bad credit rating. |
|  | On Success of the above process, ESB frames a request to update customer details in ZSS and Postilion. |
| 6 | On Successful call of T24 Response will be placed in the RAW\_LOG\_SB\_REQ to log the response in Data Base (RAW\_AUDIT\_LOGGING). |
| 7 | On Unsuccessful call of T24 Response will be placed in the ERROR\_LOG\_SB\_REQ to log the response in Database (ERR\_AUDIT\_LOGGING). |

# IIB Implementation Process Flow

## Account Opening Successful Request Response Flow

### Process Diagram



**Figure: 2 Account Opening Flow**

### Process Flow Steps

|  |  |
| --- | --- |
| **S. No** | **Activities** |
| 1. | ESB receives the request from the channel using the HTTP Router Application based on procesingcode & channelName the message will be routed to the Business Application (AccountOpeningFlow) using Business Queue (ACOP\_SB\_HTTP\_REQ)  i.e Request from channel. |
| 2 | Initial request and response are logged in the RAW\_LOG\_SB\_REQ and is insert in Database (RAW\_AUDIT\_LOGGING). |
| 3 | After logging of request ESB checks the values of the tag named “accountOpeningCheck”,The possible values are KYC\_CHECK,FCM\_CHECK and Zss/Postilion Update. |
| 4 | If the values of accountOpeningCheck tag is other than mentioned above then,ESB frames a response to Channel saying “unknown Request”. |
| 5 | If the value is “KYC\_CHECK” then the request will be routed to “KYC” Label using route to label Node for KYC process. ESB frames a request to check whether customer record exists in DWH, If customer doesn’t exists in DWH then ESB again frames a request for Eco net to check the same then the success response from DWH/Eco net will be send back to channel as response. If there is a failure in both the cases then ESB frames a failure response to channel. |
| 6 | After Framing the Eco net request and getting response from DWH or Eco net it will be logged in RAW\_LOG\_SB\_REQ and is insert in Data Base. |
| 7 | If the value is “FCM\_CHECK” then the request will be routed to “FCM” Label using route to label node for fraud customer process. ESB frames a request (Rest call) to T24 to check whether the customer is fraud or not. |
| 8 | If the T24 call is successful, then JSON response is generated. |
| 9 | T24 Response in JSON Format is logged in RAW\_LOG\_SB\_REQ and insert in Data Base (RAW\_AUDIT\_LOGGING). |
| 10 | After the Response is generated by T24 call same response will be send back to Channel with the help of HTTP Router Application. Here ESB frames the Channel Response (T24 Response). |
| 11 | T24 Response is logged in RAW\_LOG\_SB\_REQ and is insert in Database (RAW\_AUDIT\_LOGGING). |
| 12 | If the value is “Zss/Postilion Update” then the request will be routed to “Update” Label using route to label node. ESB frames a request (Rest call) to ZSS by converting REST to ISO to update the customer record in ZSS. |
| 13 | On Successful call to ZSS will return a response code as ‘00’. Then ESB frames a request to Postilion to update the same customer details in Postilion. |
| 14 | On Successful call to Postilion will return a response code as ‘00’.Then ESB frames a success response to Channel. |
| 15 | On failure either from ZSS/Postilion will leads to failure response from ESB to Channel. |
| 16 | On Unsuccessful calls of T24/ZSS/Postilion/Eco net Request and Response will be placed in the ERROR\_LOG\_SB\_REQ to log the request in Database (ERR\_AUDIT\_LOGGING). |

## Retry Process workflow

### HTTP Process Flow Diagram

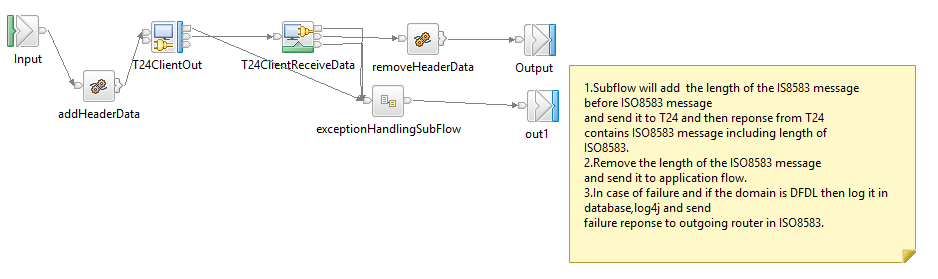


**Figure: 3 HTTP Retry for Account Opening**

### Process Flow steps

|  |  |
| --- | --- |
| **S. No** | **Activities** |
| 1 | HTTP Retry call is used to call the backend service for multiple times |
| 2 | On Successful call of backend service, response will be generated and send back to Business Application For further process |
| 3 | On Unsuccessful call of backend service it Retry for 3 times and sends error message to Business Application for further Process |

### TCP Process Flow Diagram



**Figure: 3 TCP Retry for Account Opening**

### Process Flow steps

|  |  |
| --- | --- |
| **S. No** | **Activities** |
| 1 | The TCPIP Subflow will add the length of the IS8583 message before ISO8583 message and send it to T24 and then response from T24 contains ISO8583 message including length of ISO8583. |
| 2 | The Subflow will remove the length of the ISO8583 message.  And send it to ESB application flow. |
| 3 | In case of failure and if the domain is DFDL then log it in database,log4j and send failure response to outgoing router in ISO8583. |
| 4 | TCPIP Retry call is used to call the backend service for multiple times. |
| 5 | On Successful call of backend service, response will be generated and send back to Business Application for further process. |
| 6 | On Unsuccessful call of backend service it Retry for 3 times and sends error message to Business Application for further Process. |

## Raw\_Audit Process Flow:

### Process Flow Diagram:



**Figure: 4 Raw\_Audit\_Logging Flow**

### Process Flow Steps:

|  |  |
| --- | --- |
| **S. No** | **Activities** |
|  | The RAW\_AUDIT locks the input Request and Response |
|  | After the query is executed the record is inserted into the Database. |
|  | Final Response is inserted into the Database and then Response is logged into the RAW\_LOG\_SB\_REQ. |

## HTTP Router Process Flow

### Incoming Router Process Flow Diagram:



**Figure: 5 Incoming Router Flow**

### Process Flow Steps:

|  |  |
| --- | --- |
| **S. No** | IIB receives the request from channel. |
|  | ESB receives the request from channel. |
|  | Request channel is the http request accepts the request from the channel |
|  | The request message is placed in the RAW\_LOG\_SB\_REQ to log the request in Database(RAW\_AUDIT\_LOGGING) |
|  | According to the Channel Request message fields such as channel and processing code it picks the queue name of the business flow from STWB\_ESB\_TRAN\_DETAILS table and propagates it to the business flow. |
|  | It validates the processingCode, Channel and domain of input from STWB\_ESB\_TRAN\_DETAILS if it fails to validate the error response is logged in the logs and in Data Base (ERROR\_AUDIT\_LOG). |
|  | After the validation Request message is passed through ESB. If the queue name presents in MQ manager the ESB business flow triggers otherwise the error response is sent to the front end as Unable to open queue. |
|  | If it is success or failure response will store them in RAW\_AUDIT\_LOG table or ERROR\_AUDIT\_LOG and logging into log file. |
|  | The request from channel is sent to ESB CurrencyConverter Flow |

### STEWARDBANK ESB TRANSACTION DETAILS

**Table Name: STWB\_ESB\_TRAN\_DETAILS**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Item #** | **DB Field Name** | **Field Description** | **Data Type** | **Length** | **Mandatory(Yes/No)** | **Comments** |
| 1 | PROC\_CODE | Processing Code | Varchar | 20 | Y | From input request |
| 2 | TCP\_REQ\_QNAME | TCPIP Request Queue | Varchar | 20 | N | This Request Queue Name field is used for ISO Related Application |
| 3 | TCP\_RES\_QNAME | TCPIP Response Queue | Varchar | 20 | N | This Response Queue Name field is used for ISO Related Application |
| 4 | HTTP\_REQ\_QNAME | HTTP Request Queue | Varchar | 20 | Y | This Request Queue Name field is used for REST Related Application |
| 5 | HTTP\_RES\_QNAME | HTTP Response Queue | Varchar | 20 | Y | This Response Queue Name field is used for REST Related Application |
| 6 | CHANNEL | Channel | Varchar | 20 | Y | Channel Name |
| 7 | MSGDOMAIN | Message Domain | Varchar | 20 | Y | Request Message Format |
| 8 | APPLICATION\_NAME | Name of the application | Varchar | 50 | Y | Name of the application that is create for this particular interface and committed in GIT. |
| 9 | SERVICE\_NAME | Name of the service | Varchar | 50 | Y | Name of the service to identify in the Database Table. |

### Outgoing Router Process Flow Diagram



**Figure: 6 Outgoing Router Flow**

### Process Flow Steps:

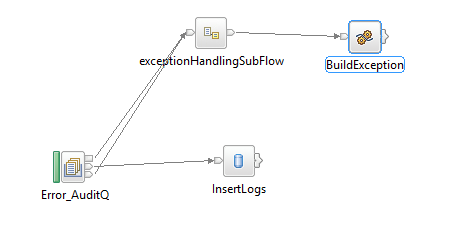
|  |  |
| --- | --- |
| **S. No** | **Activities** |
|  | The OutgoingHttpRes brings the Channel Response |
|  | Channel Response will be logged in RAW\_LOG\_SB\_REQ and insert in Data Base. |
|  | Finally Channel Response will be send to the Front End. |

## Error\_Audit Process Flow

### Process Flow Diagram:



**Figure: 7 Exception Subflow**



**Figure: 8 Error\_Audit\_Logging Flow**

### Process Flow Steps:

|  |  |
| --- | --- |
| **S. No** | **Activities** |
|  | The ERROR\_AUDIT\_Q locks the input Request. |
|  | After the query is executed the record is inserted into the Database. |
|  | Final Response is inserted into the Database and then Response is logged into the ERR\_LOG\_SB\_REQ. |

# Interface Definitions

## Request Message Definition

### Channel Request Message Structure /Schema

**Request Type:** HTTP

**Request Format:** JSON

**Request URL:** <http://IPAddress:Port/v1/stewardBank/>

### Channel Request Message Details

KYC Request:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/ Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | channel | String | Y | Channel Name |
| 2 | processingCode | String | Y | Processing Code |
| 3 | accountOpeningCheck | String | Y | Type of service call |
| 4 | national\_Id | String | Y | National id of the customer |
| 5 | requestId | String | Y | Request id |
| 6 | transactionId | String | Y | Transaction id |
| 7 | featureId | String | Y | Feature Id |
| 8 | timeStamp | String | O | Time stamp |
| 9 | channelId | String | Y | Channel id |
| 10 | languageId | String | O | Language Id |
| 11 | username | String | Y | User name |
| 12 | password | String | Y | Password |
| 13 | name | String | O | Name of a specific tag that needs to be updated. |
| 14 | value | String | O | Value of a specific tag that needs to be updated |

FCM Request:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/ Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | channel | String | Y | Channel Name |
| 2 | processingCode | String | Y | Processing Code |
| 3 | accountOpeningCheck | String | Y | Type of service call |
| 4 | customerId | String | Y | customer id of the customer |
| 5 | productId | String | Y | product id |
| 6 | currencyId | String | Y | currency id |
| 7 | activityId | String | Y | activity Id |

Update Request:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/ Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | channel | String | Y | Channel Name |
| 2 | processingCode | String | Y | Processing Code |
| 3 | accountOpeningCheck | String | Y | Type of service call |
| 4 | mobileNumber | String | Y | Mobile number of the customer |
| 5 | idNumber | String | Y | National id of the customer |
| 6 | accountNumber | String | Y | Account number |
| 7 | accountType | String | Y | Account type |
| 8 | expiryDate | String | Y | Expiry date of the account |
| 9 | actionType | String | Y | Action type |
| 10 | title | String | Y | Title |
| 11 | firstname | String | Y | First name of the customer |
| 12 | lastname | String | Y | Last name of the customer |
| 13 | middlename | String | Y | Middle name of the customer |
| 14 | birthDate | String | Y | Date of birth of the customer |
| 15 | emailAddress | String | Y | Email address of the customer |
| 16 | maritalStatus | String | Y | Marital status |
| 17 | gender | String | Y | Gender |
| 18 | identificationType | String | Y | Identification type |
| 19 | nationality | String | Y | Nationality |
| 20 | residence | String | Y | Residence |
| 21 | city | String | Y | City |
| 22 | address1 | String | Y | Address |
| 23 | address2 | String | O | Alternate address |
| 24 | country | String | Y | country |
| 25 | branch | String | Y | Branch |
| 26 | currency | String | Y | Currency |
| 27 | bankingServicesStatus | String | O | Banking service status |
| 28 | squareRegStatus | String | O | Square registration status |
| 29 | t24CustomerId | String | Y | T24 customer Id |
| 30 | detailsAmended | String | Y | Details |
| 31 | sbPostilionStatus | String | O | Steward bank postilion status |
| 32 | zssPostilionStatus | String | O | Zim switch Postilion status |

### Channel Sample Source Messages

KYC Request:

|  |
| --- |
| <request>      <channel>236</channel>      <processingCode>292000</processingCode>      <accountOpeningCheck>KYC\_CHECK</accountOpeningCheck>      <national\_Id>59029719M3</national\_Id>      <requestId>2</requestId>      <transactionId>5437654328765435</transactionId>      <featureId>search\_msisdn</featureId>      <timeStamp>1535638968307</timeStamp>      <channelId>6</channelId>      <languageId>1</languageId>      <username>Steward</username>      <password>@St3ward</password>      <data>          <param>              <name>msisdn</name>              <value>777991604</value>          </param>      </data>  </request>  FCM Request :  {      "header": {          "channel": "236",          "processingCode": "292000",          "accountOpeningCheck": "FCM\_CHECK",          "override": {              "overrideDetails": []          }      },      "body": {          "customerIds": [              {                  "customerId": "100122"              }          ],          "properties": [],          "productId": "STW.PERSONAL.CURRENT",          "currencyId": "ZWL",          "activityId": "ACCOUNTS-NEW-ARR.CHANNEL"      }  }  Update Request:  {      "channel": "236",      "processingCode": "292000",      "accountOpeningCheck": "Zss/Postilion Update",      "mobileNumber": "sample",      "idNumber": "59029719M32",      "accountNumber": "sample",      "accountType": "sample",      "expiryDate": "2707",      "actionType": "sample",      "title;": "sample",      "firstname": "sample",      "lastname": "sample",      "middlename": "sample",      "birthDate": "sample",      "emailAddress": "sample",      "maritalStatus": "sample",      "gender": "sample",      "identificationType": "sample",      "nationality": "sample",      "residence": "sample",      "city": "sample",      "address1": "sample",      "address2": "sample",      "country": "sample",      "branch": "sample",      "currency": "ZMW",      "bankingServicesStatus": "sample",      "squareRegStatus": "sample",      "t24CustomerId": "sample",      "detailsAmended": "N",      "sbPostilionStatus": "sample",      "zssPostilionStatus": "sample"  } |

### T24 Request Message Structure /Schema

**Request Type:** HTTP

**Request Format:** JSON

**Request URL:**http://IPAddress:Port/stwbc/api/v1.0.0/stwb/enquiries/currencies/details

### T24 Response Message Details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/ Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | customerId | String | Y | Customer id of the customer |
| 2 | productId | String | Y | Id of the product selected by customer |
| 3 | currencyId | String | Y | Currency Id of the country that customer belongs to. |
| 3 | activityId | String | Y | Activity id. |

### T24 Sample Source Messages

|  |
| --- |
| {     "header":{            "override":{           "overrideDetails":[           ]        }     },     "body":{        "customerIds":[           {              "customerId":"100121"           }        ],        "properties":[        ],        "productId":"STW.PERSONAL.CURRENT",        "currencyId":"ZWL",        "activityId":"ACCOUNTS-NEW-ARR.CHANNEL"     }  } |

## Response Message Definition

### Channel Response Message Structure/Schema

**Response Type:** JSON

### Channel Response Message Details

When KYC found in DWH:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | channel | String | Y | Channel Name |
| 2 | processingCode | String | Y | Processing code of the interface |
| 2 | accountOpeningCheck | String | Y | Account opening check |
| 3 | status | String | Y | Status of the response |
| 3 | responseCode | String | Y | Response code |
| 3 | firstname | String | Y | Firstname of the customer |
| 4 | lastname | String | Y | Last name of the customer |
| 4 | dob | String | Y | Date of birth |
| 4 | gender | String | Y | Gender |
| 4 | address | String | Y | Address of the custmer |

When KYC not found in DWH:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | code | String | Y | Response code |
| 2 | desc | String | Y | Description of the response |
| 2 | name | String | Y | Name of the field |
| 3 | value | String | Y | Value of the field |

FCM Response to Channel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/ Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | transactionStatus | String | Y | Header Tag |
| 2 | audit | String | Y | Audit Tag |
| 3 | T24\_time | String | Y | Time for T24 |
| 3 | parse\_time | String | Y | Time of parsing. |
| 2 | page\_start | String | Y | Starting page. |
| 2 | page\_token | String | Y | Token number. |
| 2 | total\_size | String | Y | Total size of response. |
| 2 | page\_size | String | Y | Size of the response pages. |
| 1 | body | String | Y | Body Tag |
| 2 | Markets\* | String | Y | Markets details |
| 3 | market | String | Y | Market name |
| 3 | sellRate | String | Y | Selling Rate |
| 3 | buyRate | String | Y | Buying Rate |
| 2 | displayName | String | Y | Display name of the currency. |
| 2 | currencyId | String | Y | Currecny Id |

Update Response to Channel:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item #** | **Tag Name/ Field Name** | **Data Type** | **Mandatory(Yes/No)** | **Comments** |
| 1 | status | String | Y | Status of the response. |
| 2 | responseCode | String | Y | Response code. |
| 3 | Account | String | Y | Account number of the customer. |
| 3 | mobileNumber | String | Y | Mobile number of the customer. |
| 2 | sbPostilionResponseCode | String | Y | Steward bank postilion response code. |
| 2 | zssPostilionResponseCode | String | Y | Zss postilion response code. |

### Channel Sample Response Messages

|  |
| --- |
| KYC Response to channel :  When KYC found in DWH :    <AccountOpeningResponse>  <header>  <channel>236</channel>  <processingCode>292000</processingCode>  <accountOpeningCheck>KYC\_CHECK</accountOpeningCheck>  <status>SUCCESS</status>  <responseCode>200</responseCode>  </header>  <body>  <firstname>JOSIAH</firstname>  <lastname>NGWENDERE </lastname>  <dob>19780926</dob>  <gender></gender>  <address>2810KUWADZANA4</address>  </body>  </AccountOpeningResponse>  When KYC not found in DWH :  <acknowledgment>  <status>  <code>SC0000</code>  <desc>SUCCESS</desc>  </status>  <data>  <param>  <name>FIRST NAME</name>  <value>PHILLIPA</value>  </param>  <param>  <name>LAST NAME</name>  <value>KANJANDA</value>  </param>  <param>  <name>DOB</name>  <value>09-01-1968</value>  </param>  <param>  <name>GENDER</name>  <value>M</value>  </param>  <param>  <name>EMAIL</name>  <value>NULL</value>  </param>  <param>  <name>STATUS</name>  <value>REGISTERED</value>  </param>  <param>  <name>ICCID</name>  <value/>  </param>  <param>  <name>ALTERNATE CONTACT NUMBER</name>  <value/>  </param>  <param>  <name>ADDRESS</name>  <value>MWOYOWESHUMBA VILLAGE CHIEF MUTASAMOYOWESHUMBA PRIMARY</value>  </param>  <param>  <name>VILLAGE</name>  <value/>  </param>  <param>  <name>PROVINCE</name>  <value>MANICALAND</value>  </param>  <param>  <name>CITY</name>  <value>WATSOMBA</value>  </param>  <param>  <name>DISTRICT</name>  <value/>  </param>  <param>  <name>CUSTOMER TYPE</name>  <value>PRIVATE NATIONAL</value>  </param>  <param>  <name>APPLICATION NUMBER</name>  <value>NULL</value>  </param>  <param>  <name>DOC ID</name>  <value>2</value>  </param>  <param>  <name>DOC TYPE</name>  <value>National ID</value>  </param>  <param>  <name>DOC NUMBER</name>  <value>50053402W50</value>  </param>  <param>  <name>CREATE USER</name>  <value>NULL</value>  </param>  <param>  <name>CREATE USER MSISDN</name>  <value>NULL</value>  </param>  <param>  <name>CREATE DATE</name>  <value>2011-02-23 20:02:03.0</value>  </param>  <param>  <name>ACTIVATION DATE</name>  <value>2011-02-23 20:02:03.0</value>  </param>  <param>  <name>COMPANY PHYSICAL ADDRESS</name>  <value/>  </param>  <param>  <name>COMPANY NAME</name>  <value/>  </param>  <param>  <name>INDUSTRY TYPE</name>  <value/>  </param>  <param>  <name>COMPANY REG NO</name>  <value/>  </param>  <param>  <name>VERIFICATION STATUS</name>  <value>CUSTOMER DETAILS VERIFIED</value>  </param>  <param>  <name>KIN FIRST NAME</name>  <value/>  </param>  <param>  <name>KIN SUR NAME</name>  <value/>  </param>  <param>  <name>KIN CONTACT NUMBER</name>  <value/>  </param>  <param>  <name>TITLE</name>  <value>Miss</value>  </param>  <param>  <name>PASSPORT EXPIRATION DATE</name>  <value/>  </param>  <param>  <name>MARITAL STATUS</name>  <value/>  </param>  <param>  <name>SUBSCRIPTION LIMIT</name>  <value>4</value>  </param>  <param>  <name>CUSTOMER CREATION DATE</name>  <value>2016-01-18 08:57:48.0</value>  </param>  </data>  </acknowledgment>  FCM Response to Channel :  {  "header": {  "transactionStatus": "Live",  "audit": {  "T24\_time": 12395,  "responseParse\_time": 12,  "requestParse\_time": 23565  },  "aaaId": "AAACT20247XKCM1NC3",  "status": "success"  },  "body": {  "arrangementActivity": {  "arrangementId": "AA202470X866",  "activityId": "ACCOUNTS-NEW-ARR.CHANNEL",  "productId": "STW.PERSONAL.CURRENT",  "customerIds": [  {  "customerId": "100121"  }  ],  "currencyId": "ZWL",  "effectiveDate": "2020-09-03"  }  }  }  Update Response to Channel :  {  "AccountOpeningResponse": {  "header": {  "status": "SUCCESS",  "responseCode": "200"  },  "body": {  "Account": "1008540671",  "mobileNumber": "1008540671",  "sbPostilionResponseCode": "00",  "zssPostilionResponseCode": "00"  }  }  } |

### Channel Error Response Message

**Error Response:**

{

"AccountOpening": {

"header": {

"channel": "236",

"processingCode": "292000",

"status": "FAILED",

"responseCode": "000"

},

"responseBody": {

"source": "ESB",

"applicationName": "StewardBankAccountOpeningApp",

"applicationErrorCode": " Error code displays here ",

"message": " Error message displays here "

}

}

}

### Data Base Error Response Codes

001 = "Fatal Exception";

002 = "Recoverable Exception";

003 = "Configuration Exception";

004 = "Security Exception";

005 = "Parser Exception";

006 = "Conversion Exception";

007 = "Data Base Exception";

008 = "User Exception";

009 = "Cast Exception";

010 = "Message Exception";

011 = "SQL Exception";

012 = "Socket Exception";

013 = "Socket Timeout Exception";

014 = "Unknown Exception";

015 = "Failure";

# Logging Mechanism

## Insert into RAW\_AUDIT\_TABLE

**Table Name: RAW\_AUDIT\_TABLE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Item #** | **DB Field Name** | **Field Description** | **Data Type** | **Length** | **Mandatory(Yes/No)** | **Comments** |
| 1 | MSGID | MessageID | Varchar | 100 | Y | From input request |
| 2 | LOGGING\_TIME | Logging Time | TimeStamp | 6 | Y | Name |
| 3 | MESSAGE | Message(Request/Response) | Clob | - | Y | Request from channel |
| 4 | MESSAGETYPE | Type of Message(Request/Response) | Varchar | 100 | Y | Type of message either Request or Response |
| 5 | APPNAME | Application Name | Varchar | 100 | Y | Name of the Application |
| 6 | BROKER | Broker Name | Varchar | 100 | Y | Broker Name |
| 7 | TIME\_LOCAL\_TRANSACTION | Transaction Time | Varchar | 20 | Y | Time of the Transaction |
| 8 | DATE\_LOCAL\_TRANSACTION | Transaction Date | Varchar | 20 | Y | Date of the Transaction |
| 9 | RETRIEVAL\_REFERENCE\_N | Retrieval Reference Number | Varchar | 20 | Y |  |

## Insert into ERROR\_AUDIT\_TABLE

**Table Name: ERROR\_AUDIT\_TABLE**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Item #** | **DB Field Name** | **Field Description** | **Data Type** | **Length** | **Mandatory(Yes/No)** | **Comments** |
| 1 | MSGID | MessageID | Varchar | 50 | Y | From input request |
| 2 | LOGGING\_TIME | Application Name | Timestamp | 50 | Y | Time of the logging |
| 3 | MESSAGE | Message Type(Request/Response) | Clob | 50 | Y | Request of the Message |
| 4 | MESSAGETYPE | Message Type | Varchar | 50 | Y | Type of message either Request or Response |
| 5 | APPNAME | Application Name | Varchar | 4000 | Y | Name of the Application |
| 6 | BROKER | Broker Name | Varchar | - | Y | Name of the Broker |
| 7 | ERRORDESCRIPTION | Exception | Clob | 4000 | Y | Exception Information |
| 8 | TIME\_LOCAL\_TRANSACTION | Transaction Time | Varchar | 20 | Y | Time of the Transaction |
| 9 | DATE\_LOCAL\_TRANSACTION | Transaction Date | Varchar | 20 | Y | Date of the Transaction |
| 10 | RETRIEVAL\_REFERENCE\_NUMBER | Retrieval Reference Number | Varchar | 20 | Y | Retrieval Reference Number |
| 11 | BORKER\_ERROR\_CODE | Broker Error Code | Varchar |  | Y | Error code generate by Broker |
| 12 | USER\_DEFINE\_ERROR\_CODE | User Defined Error Code | Varchar |  | Y | User Defined Code |

## File Based Logging

### Log4j

As part of auditing the request we have to use Log4j mechanism in the application to log the request in the file.

File Path: To be specified, while deploying the application