Purchase Orders & PO Changes Integration into your ERP

# Introduction

## General API Setup Considerations

[Get Started with the Coupa API](https://success.coupa.com/Integrate/Technical_Documentation/API/Get_Started)

[API Key Security](https://success.coupa.com/Integrate/Technical_Documentation/API/Get_Started/API_Key_Security) (will eventually get deprecated)

[Open Connect API Access](https://success.coupa.com/Support/Releases/29/New_Features/BSM_Platform/Core_Platform/Open_Connect_API_Access) (R29 onward)

[JSON/XML](https://success.coupa.com/Integrate/Technical_Documentation/API/Get_Started/Differences_between_XML_and_JSON_in_Coupa)

[GraphQL](https://success.coupa.com/Support/Releases/30/New_Features/BSM_Platform/Coupa_Platform/z_Other_Enhancements) (R30 onward)

## Limited Payload - Fields & API Filters

Coupa's API returns a lot of data by default (for example: full objects for associated objects). The API return payloads can be very large and therefore slow. This can be a problem for customers that do not need the extraneous data not to mention the unnecessary consumption of resources.

To make things easier, Coupa has the concept of the “fields” parameter & API Filters that return a limited JSON or XML response instead of the entire schema, and all associations, for an object.

[API Response Filters](https://success.coupa.com/Integrate/KB/API_Response_Filters)

[API: Using the new API query parameter "fields", Alternative to "return object"](https://success.coupa.com/Integrate/KB/API%3A_Using_the_new_API_query_parameter_%22fields%22%2C_Alternative_to_%22return_object%22)

## Use cases described in this Article

The assumption in this document is that the Purchase Order number in the ERP is driven by the Coupa Purchase Order Number

This Article describes the 3 different options to Integrate Coupa Purchase Orders into your ERP(for both creation and updates).

These options will change the way you can monitor your integrations from Coupa

* [**Option 1**: Simple monitoring based on custom field(s) you define on the Order Header](#_c78h7p79oglq)
* [**Option 2**: Advanced monitoring using Integration History records](#_qsf3yk6fikh)
* [**Option 3**: Leverage full Coupa Integrations monitoring](#_wbmd8fic8ha6)

For the 3 options, the Coupa POs are pushed to the ERP based on the Export Flag.

In case of an error, in the Integration of a PO, a manual change will be required on the PO in the Coupa UI: this change will reset the Export Flag, and the PO will therefore be considered in the next run

We recommend to have the following option set (request to Coupa Support)

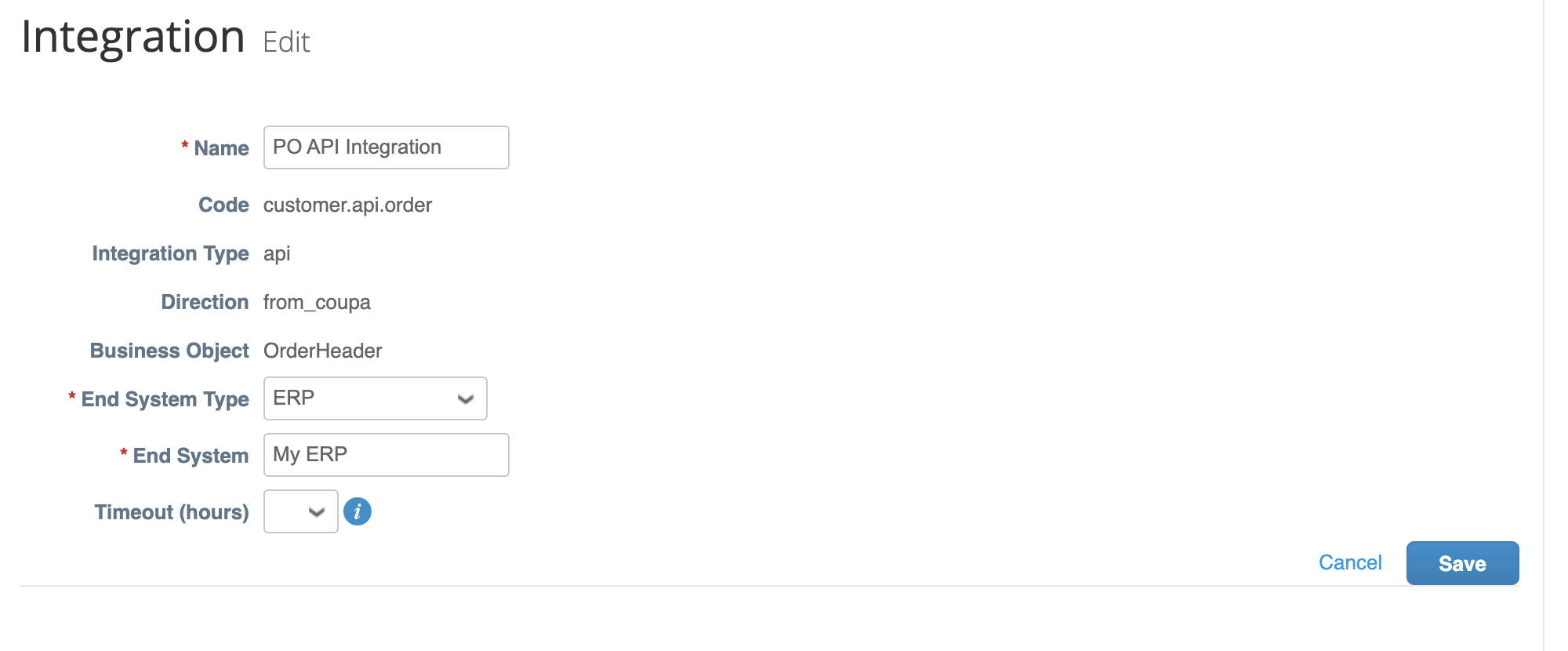
**Reset PO last exported at for every change**

*This setting only applies to the UI, changes made with API don’t change the Exported status.*

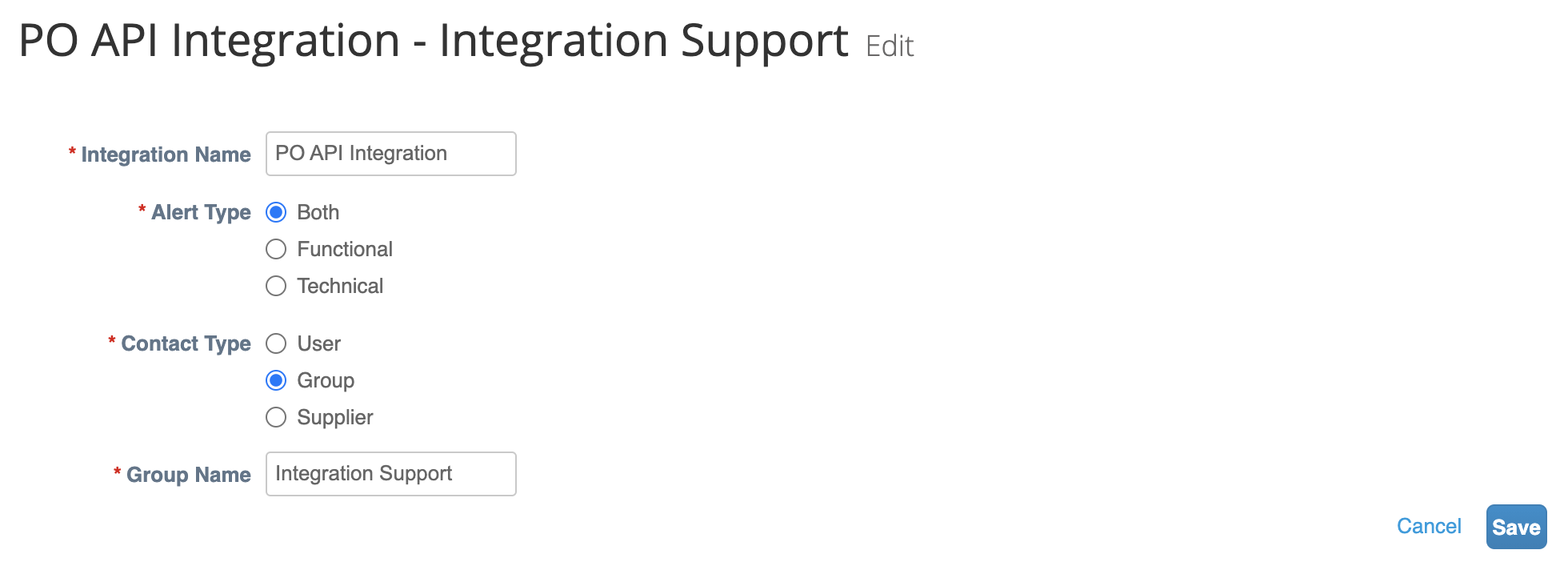
## Create dedicated Integration and Contact for API

For option 2 & 3, you will need to create:

* A dedicated Integration for each API Orchestration you implement.



* One or several Integration contact(s) for each integration, that will be alerted for any failed integration



We recommend to have the following option set (request to Coupa Support)

**Enable link to Integration History by Document Type**

*This setting adds a link in the setup page to view full Integration History per document type*

# Option 1: Simple monitoring based on custom field(s)

## Description

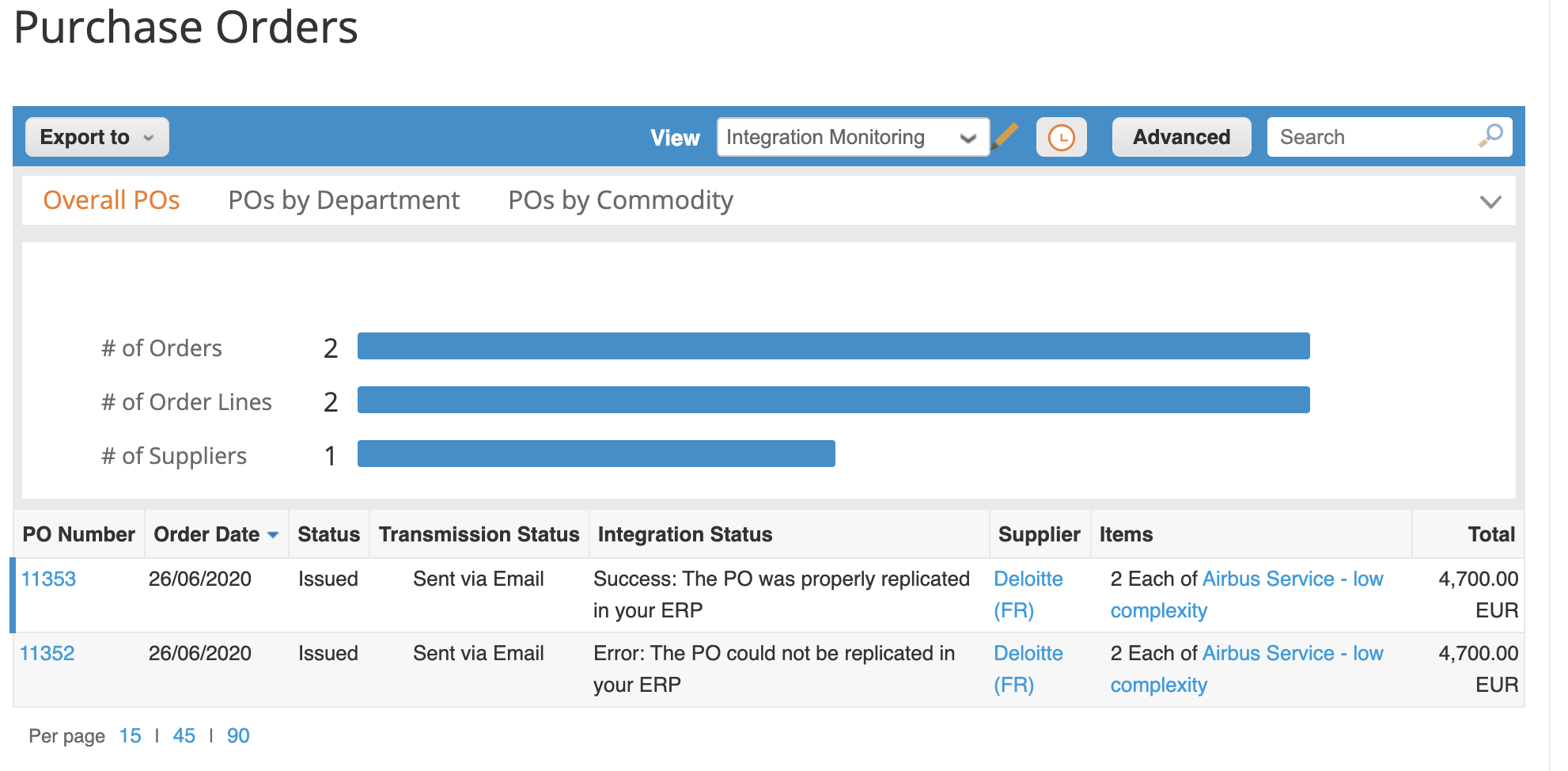
In this scenario, for each PO we:

* Add the current Integration status in one or several PO header custom field(s)

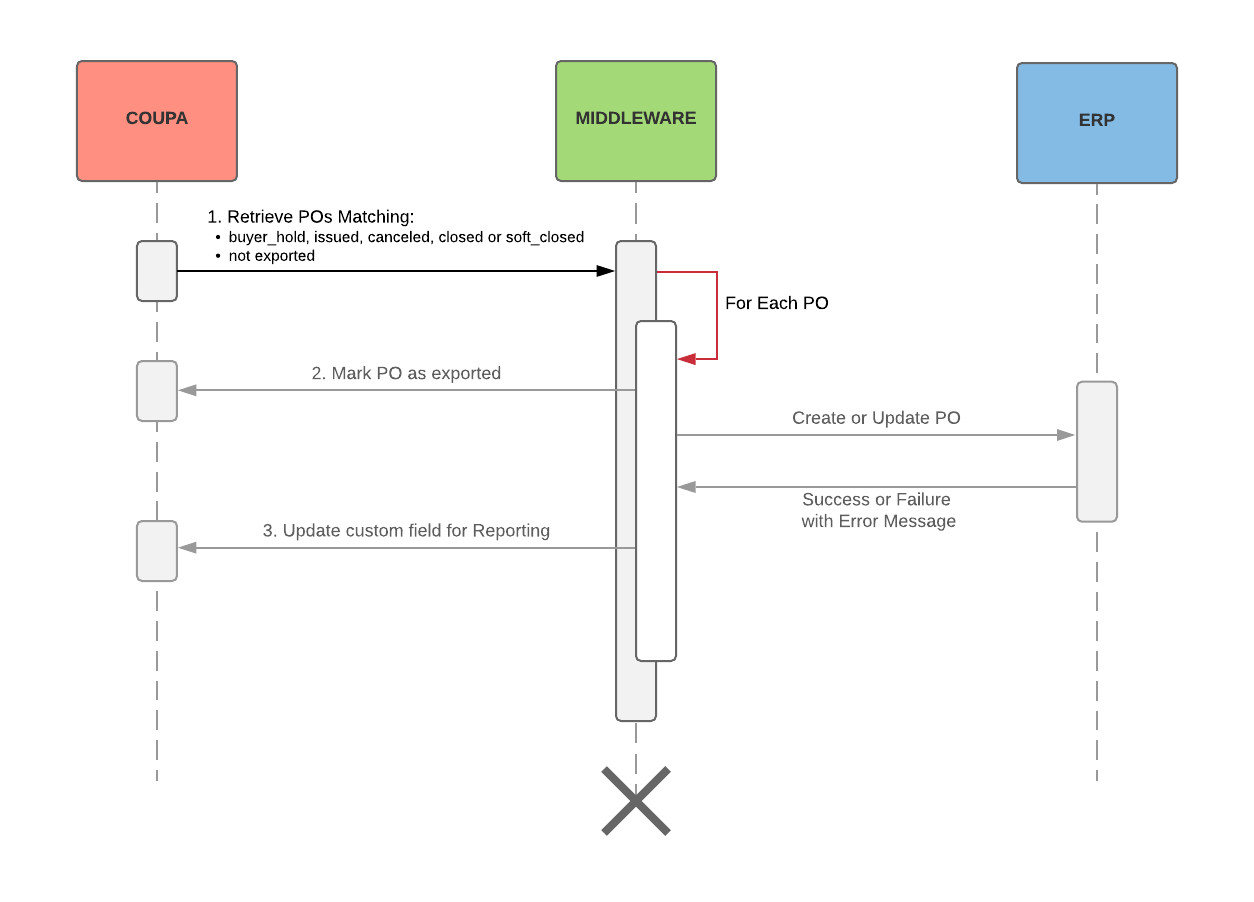
In our example the custom field cf-integration-status was created with name Integration Status

You will use the Standard PO Data Table to follow up the Integration status of your documents.

**https://*<your instance hostname>*/order\_headers**



## Orchestration Diagram



## Steps / API calls details

|  |  |
| --- | --- |
| **Step 1** | Get the list and details of the Coupa Purchase Orders to create/update in the ERP.  Selection criteria includes the **Export Flag** and **PO Status.**  Query parameter ***show\_deleted\_lines*** can be necessary to handle the PO update in the ERP |
| **Method** | GET |
| **API** | https://*<your instance hostname>*/api/purchase\_orders/ |
| **Query Params** | filter=*<your API filter name>*  offset=10  exported=false  show\_deleted\_lines=true\*  status[in]=issued,canceled,closed |
| **Sample URL** | https://*<your instance hostname>*/api/purchase\_orders?filter=*<your API filter name>*&offset=10&exported=false&show\_deleted\_lines=true&status[in]=issued,canceled,closed |
| **Query Body sample** | N/A |
| **Response Body sample** |  |

\* = show\_deleted\_lines=true is for PO changes

|  |  |
| --- | --- |
| **Step 2** | Mark individual PO as exported |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>* |
| **Query Params** | exported=true  fields=["id","exported"] |
| **Sample URL** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>*?exported=true&fields=["id","exported"] |
| **Query Body sample** | N/A \* |
| **Response Body sample** | {  "id": *<Purchase Order id>*,  "exported": true  } |

* \* For this call you do not need a payload if you include “?exported=true” in the URL

That’s where the creation/update of the PO in the ERP happens

|  |  |
| --- | --- |
| **Step 3** | Update a Custom Fields for Reporting |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>* |
| **Query Params** | fields=["id","exported",{ "custom\_fields": {} }] |
| **Sample URL** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>*?fields=["id","exported",{ "custom\_fields": {} }] |
| **Query Body sample** | { "custom-fields": {  "*<your custom-field name>*": "Success: The PO was properly replicated in your ERP"  }}  OR  { "custom-fields": {  "*<your custom-field name>*": "Error: The PO was properly replicated in your ERP"  }} |
| **Response Body sample** | { "id": *<Purchase Order id>*,  "exported": true,  "custom-fields": {  "*<your custom-field name>*": "Success: The PO was properly replicated in your ERP"  }}  OR  { "id": *<Purchase Order id>*,  "exported": true,  "custom-fields": {  "*<your custom-field name>*": "Error: The PO was properly replicated in your ERP"  }} |

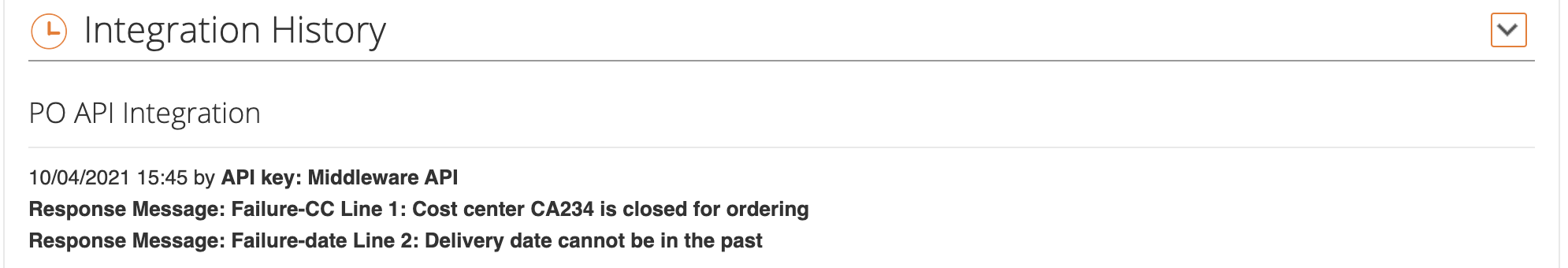
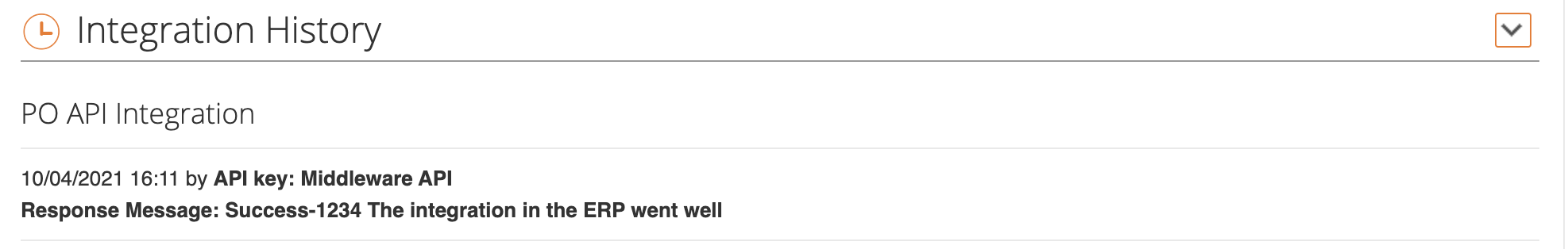
# Option 2: Advanced monitoring using Integration History

## Description

In this scenario, for each PO we:

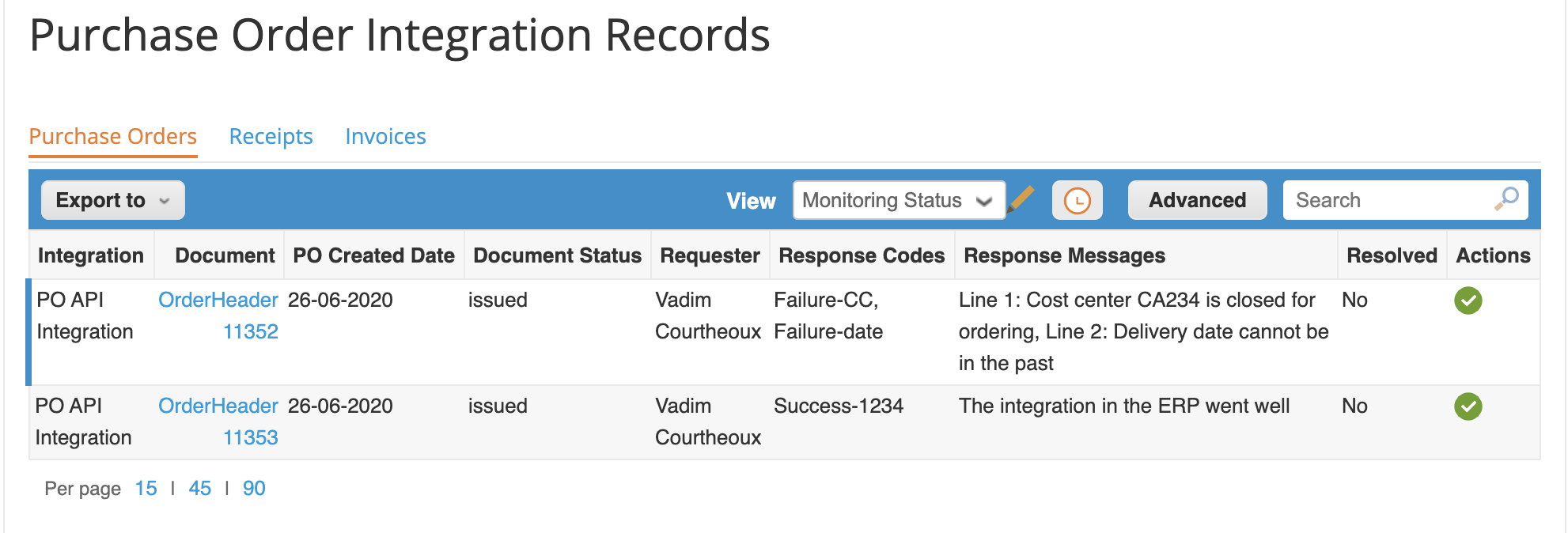
* Resolve the previous Integration History Record for the document
* Create an Integration History Record
* Create an Alert to the Integration Contact in case of an error

Each document includes the Integration History details

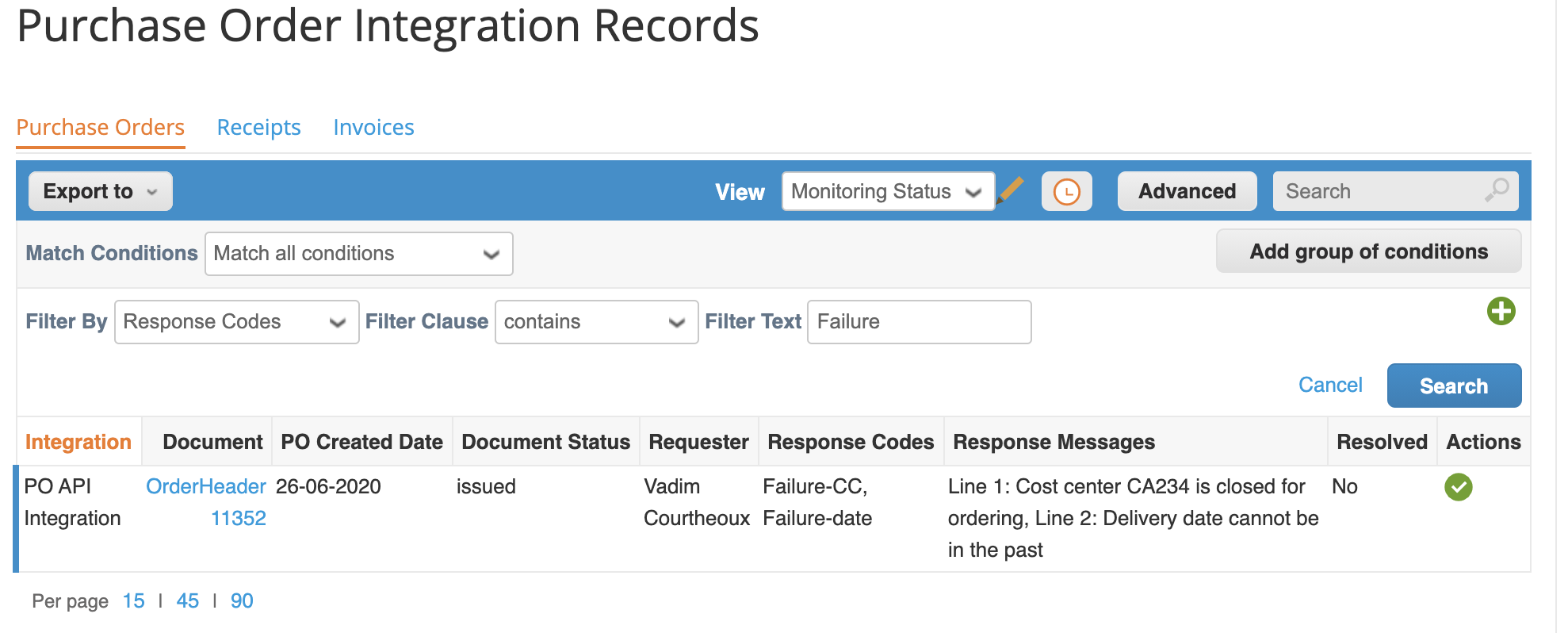


You will use the Standard PO Integration History Data Table to follow up the Integration status of your documents

**https://*<your instance hostname>*/integration\_history\_records/purchase\_orders**

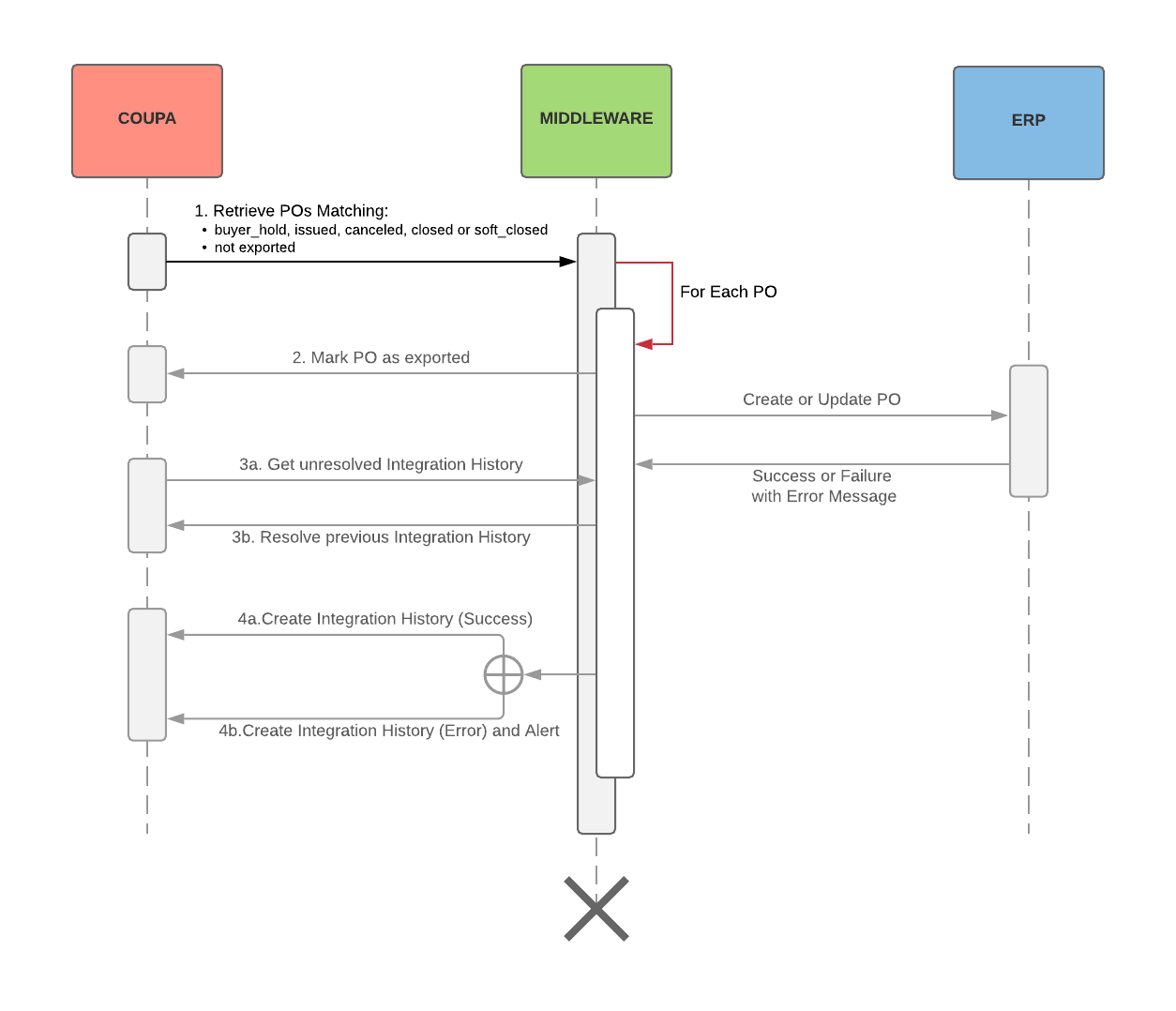


You can use a filter on Response Code to differentiate between documents successfully replicated and documents that failed



## 

## Orchestration Diagram



## Steps / API calls details

|  |  |
| --- | --- |
| **Step 1** | Get the list and details of the Coupa Purchase Orders to create/update in the ERP.  Selection criteria includes the **Export Flag** and **PO Status.**  Query parameter ***show\_deleted\_lines*** can be necessary to handle the PO update in the ERP |
| **Method** | GET |
| **API** | https://*<your instance hostname>*/api/purchase\_orders/ |
| **Query Params** | filter=*<your API filter name>*  offset=10  exported=false  show\_deleted\_lines=true  status[in]=buyer\_hold,issued,canceled,closed,soft\_closed |
| **Sample URL** | https://*<your instance hostname>*/api/purchase\_orders?filter=*<your API filter name>*&offset=10&exported=false&show\_deleted\_lines=true&status[in]=buyer\_hold,issued,canceled,closed,soft\_closed |
| **Query Body sample** | N/A |
| **Response Body sample** |  |

|  |  |
| --- | --- |
| **Step 2** | Mark individual PO as exported |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>* |
| **Query Params** | exported=true  fields=["id","exported"] |
| **Sample URL** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>*?exported=true&fields=["id","exported"] |
| **Query Body sample** | N/A |
| **Response Body sample** | {  "id": *<Purchase Order id>*,  "exported": true  } |

That’s where the creation/update of the PO in the ERP happens

|  |  |
| --- | --- |
| **Step 3a** | Get unresolved Integration History Record |
| **Method** | GET |
| **API** | https://*<your instance hostname>*/api/integration\_history\_records |
| **Query Params** | resolved=false  document-type=OrderHeader  document-id=*<Purchase Order id>*  fields=["id","document-id","status","resolved"] |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_history\_records?resolved=false&document-type=OrderHeader&document-id=*<Purchase Order id>*&fields=["id","document-id","status","resolved"] |
| **Query Body sample** | N/A |
| **Response Body sample** | [  {  "id": *<Old Integration History Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Error",  "resolved": false  }  ] |

|  |  |
| --- | --- |
| **Step 3b** | Resolve previous Integration History Record |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_history\_records/*<Old Integration History Record id>*/resolve |
| **Query Params** | fields=["id","document-id","status","resolved"] |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_history\_records/*<Old Integration History Record id>*/resolve?fields=["id","document-id","status","resolved"] |
| **Query Body sample** | N/A |
| **Response Body sample** | [  {  "id": *<Old Integration History Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Error",  "resolved": true  }  ] |

|  |  |
| --- | --- |
| **Step 4a** | Create Integration History (Success) |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_history\_records |
| **Query Params** | fields=["id","document-id","status"] |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_history\_records?fields=["id","document-id","status"] |
| **Query Body sample** | {  "document-type": "OrderHeader",  "document-id": *<Purchase Order id>*,  "document-status":"*<Purchase Order status>*",  "contact-alert-type": "Functional",  "status": "**Success**",  "integration": {"code":"*<Customer Integration id>*"},  "responses": [  {  "response-code": "Success-1234",  "response-message": "The integration in the ERP went well"  }  ]  } |
| **Response Body sample** | {  "id": *<New Integration History Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Success"  } |

|  |  |
| --- | --- |
| **Step 4b** | Create Integration History (Error) and alert to Integration Contact |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_history\_records/create\_alert |
| **Query Params** | fields=["id","document-id","status"] |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_history\_records/create\_alert?fields=["id","document-id","status"] |
| **Query Body sample** | {  "document-type": "OrderHeader",  "document-id": *<Purchase Order id>*,  "document-status":"*<Purchase Order status>*",  "contact-alert-type": "Functional",  "status": "**Error**",  "integration": {"code":"*<Customer Integration id>*"},  "responses": [  {  "response-code": "Failure-CC",  "response-message": "Line 1: Cost center CA234 is closed for ordering"  },  {  "response-code": "Failure-date",  "response-message": "Line 2: Delivery date cannot be in the past"  }  ]  } |
| **Response Body sample** | {  "id": *<New Integration History Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Error"  } |

## 

# 

# Option 3: Leverage full Coupa Integrations monitoring

## Description

In this scenario, we create an Integration run that tracks

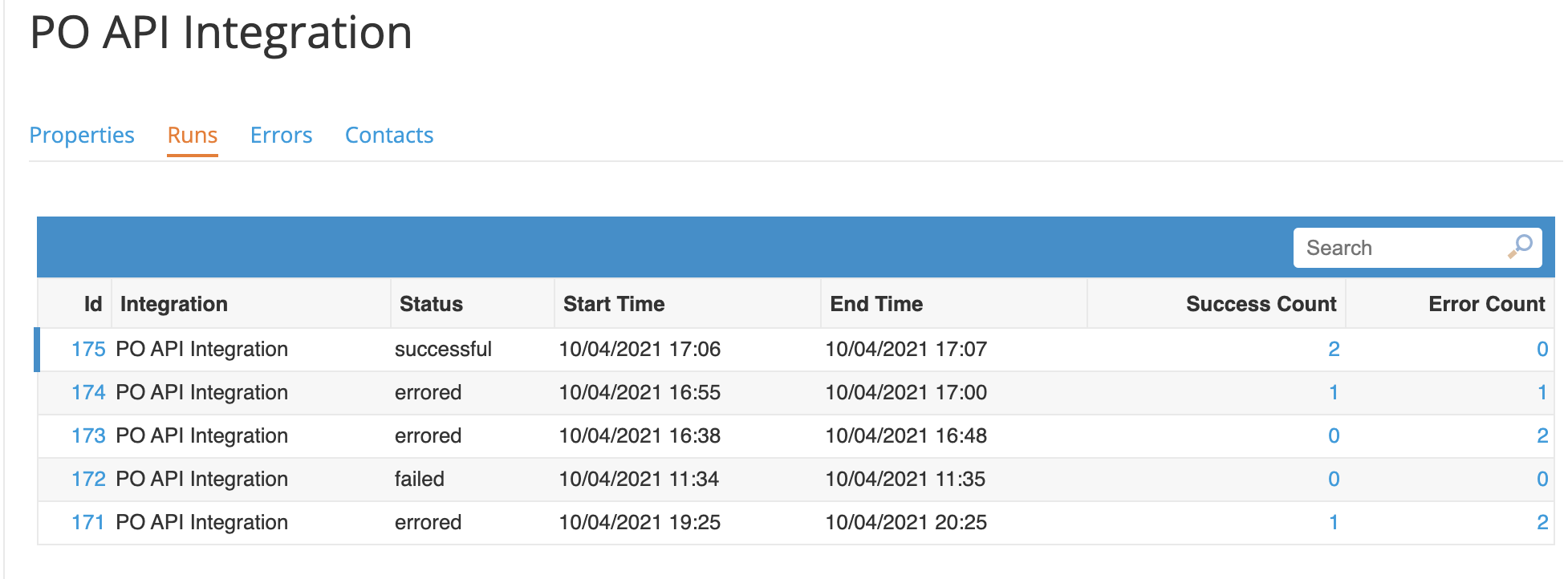
* The status of the Integration (pending/started/errored/successful/failed)
* The total number of PO processed
* The number of Success and Errors
* The list of Integration Errors and their statuses (resolved or not)

For each PO we:

* Resolve the previous Integration History Record for the document
* Create an Integration History Record
* In case of an error, create an Integration Error and an Alert to the Integration Contact

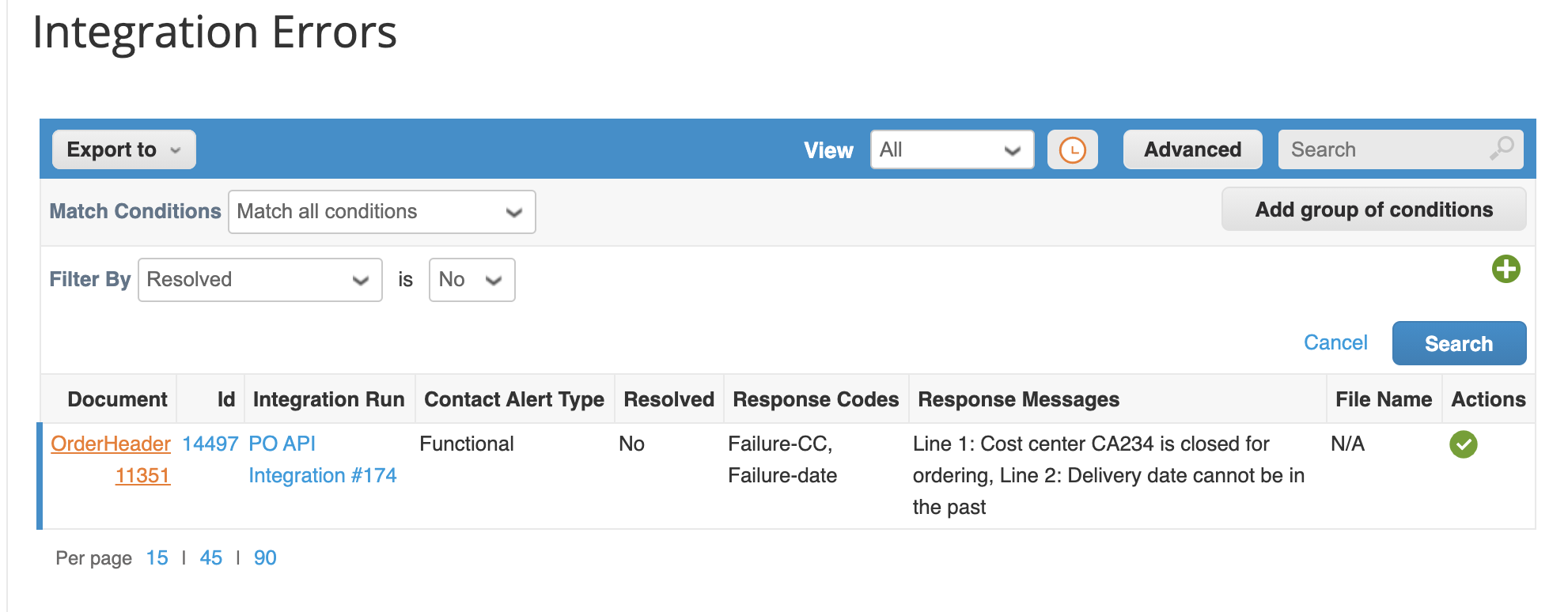
You will be able to monitor all Integration Runs for your Integration

**https://*<your instance hostname>*/integrations/*<your integration id>*/integration\_runs**



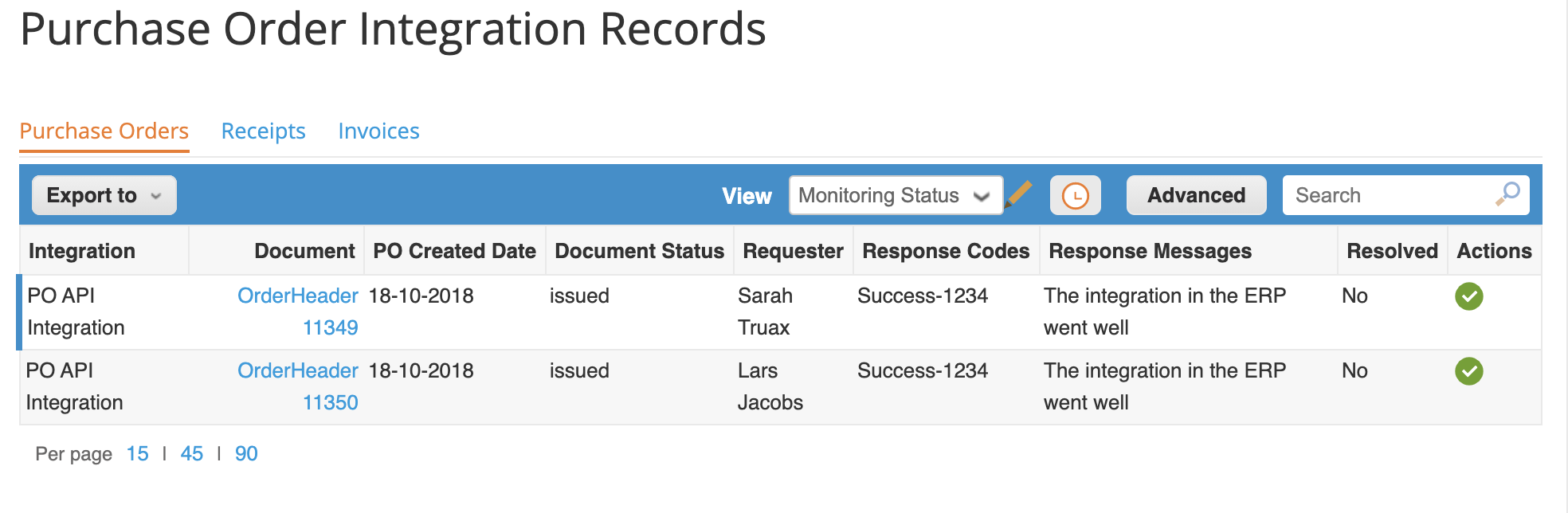
You will use the Standard PO Integration Error Data Table to list all POs with Integration Error pending resolution

**https://*<your instance hostname>*/integration\_errors**

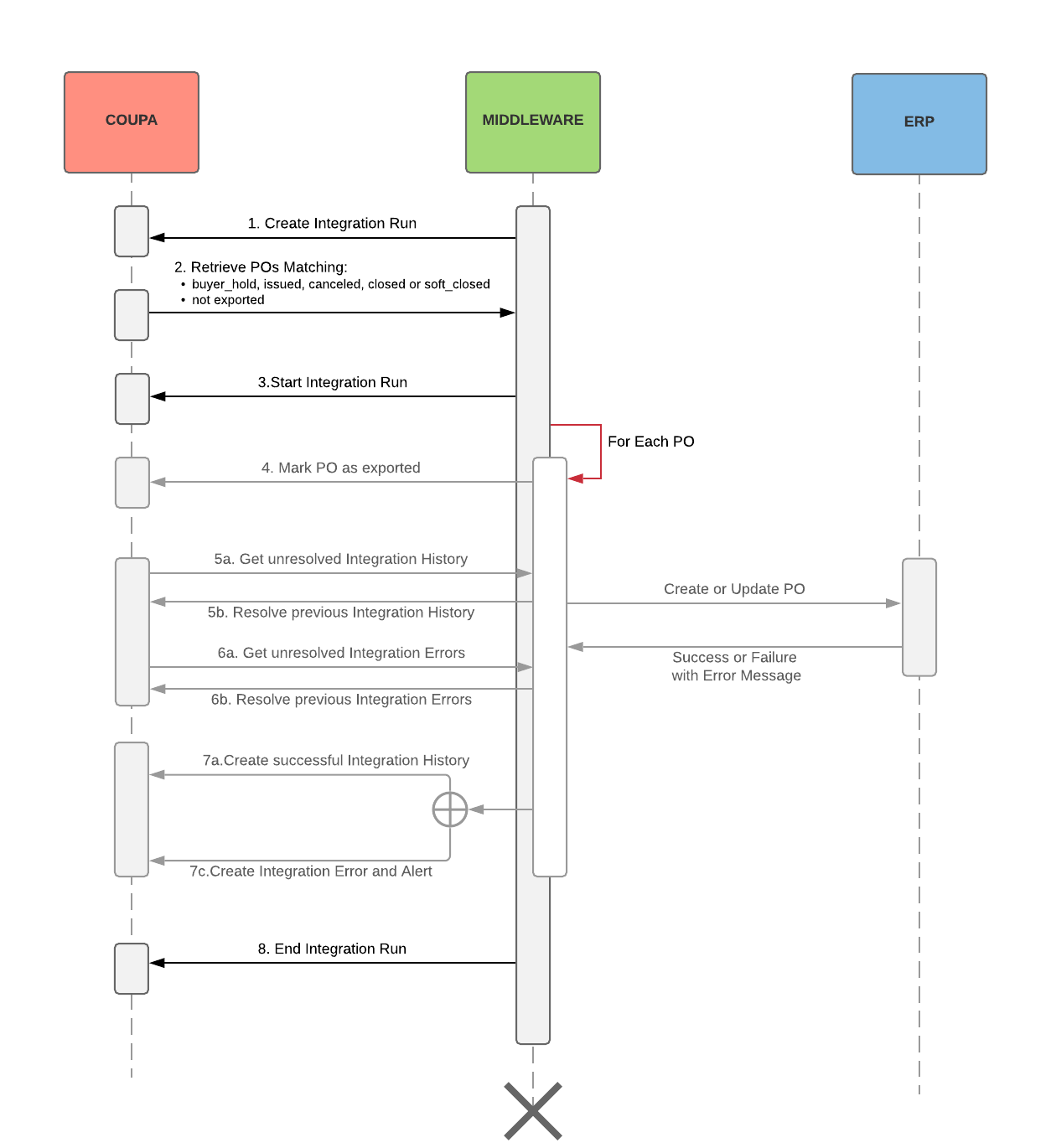


You will use the Standard PO Integration History Data Table to list all POs successfully Integrated

**https://*<your instance hostname>*/integration\_history\_records/purchase\_orders**



## Orchestration Diagram



## Steps / API calls details

|  |  |
| --- | --- |
| **Step 1** | Create Integration Run |
| **Method** | POST |
| **API** | https://*<your instance hostname>*/api/integration\_runs |
| **Query Params** | N/A |
| **Query Body sample** | {  "integration": {  "code": "{{Integration\_Code}}"  }} |
| **Response Body sample** | {  "id": *<Integration Run ID>*,  ...  } |

|  |  |
| --- | --- |
| **Step 2** | Get the list and details of the Coupa Purchase Orders to create/update in the ERP.  Selection criteria includes the **Export Flag** and **PO Status.**  Query parameter ***show\_deleted\_lines*** can be necessary to handle the PO update in the ERP |
| **Method** | GET |
| **API** | https://*<your instance hostname>*/api/purchase\_orders/ |
| **Query Params** | filter=*<your API filter name>*  offset=10  exported=false  show\_deleted\_lines=true  status[in]=buyer\_hold,issued,canceled,closed,soft\_closed |
| **Sample URL** | https://*<your instance hostname>*/api/purchase\_orders?filter=*<your API filter name>*&offset=10&exported=false&show\_deleted\_lines=true&status[in]=buyer\_hold,issued,canceled,closed,soft\_closed |
| **Query Body sample** | N/A |
| **Response Body sample** |  |

|  |  |
| --- | --- |
| **Step 3** | Start Integration Run |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_runs/*<Integration Run ID>*/run |
| **Query Params** | N/A |
| **Query Body sample** | {  "total\_records": *<Total Number of POs from Step 2>*  } |
| **Response Body sample** | {  "id": *<Integration Run ID>*,  ...  } |

|  |  |
| --- | --- |
| **Step 4** | Mark individual PO as exported |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>* |
| **Query Params** | exported=true  fields=["id","exported"] |
| **Sample URL** | https://*<your instance hostname>*/api/purchase\_orders/*<Purchase Order id>*?exported=true  fields=["id","exported"] |
| **Query Body sample** | N/A |
| **Response Body sample** | {  "id": *<Purchase Order id>*,  "exported": true  } |

That’s where the creation/update of the PO in the ERP happens

|  |  |
| --- | --- |
| **Step 5a** | Get unresolved Integration History Record |
| **Method** | GET |
| **API** | https://*<your instance hostname>*/api/integration\_history\_records |
| **Query Params** | resolved=false  document-type=OrderHeader  document-id=*<Purchase Order id>*  fields=["id","document-id","status","resolved"] |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_history\_records?resolved=false&document-type=OrderHeader&document-id=*<Purchase Order id>*&fields=["id","document-id","status","resolved"] |
| **Query Body sample** | N/A |
| **Response Body sample** | [  {  "id": *<Old Integration History Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Error",  "resolved": false  }  ] |

|  |  |
| --- | --- |
| **Step 5b** | Resolve previous Integration History Record |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_history\_records/*<Old Integration History Record id>*/resolve |
| **Query Params** | fields=["id","document-id","status","resolved"] |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_history\_records/*<Old Integration History Record id>*/resolve?fields=["id","document-id","status","resolved"] |
| **Query Body sample** | N/A |
| **Response Body sample** | [  {  "id": *<Old Integration History Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Error",  "resolved": true  }  ] |

|  |  |
| --- | --- |
| **Step 6a** | Get unresolved Integration Error Record |
| **Method** | GET |
| **API** | https://*<your instance hostname>*/api/integration\_errors |
| **Query Params** | resolved=false  document-type=OrderHeader  document-id=*<Purchase Order id>* |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_errors?resolved=false&document-type=OrderHeader&document-id=*<Purchase Order id>* |
| **Query Body sample** | N/A |
| **Response Body sample** | [  {  "id": *<Old Integration Error Record id>*,  "document-type": ”OrderHeader”,  "document-id": *<Purchase Order id>*,  "status": "Error",  "resolved": false  ...  }  ] |

|  |  |
| --- | --- |
| **Step 6b** | Resolve previous Integration History Record |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_errors/*<Old Integration Error Record id>*/resolve |
| **Query Params** | N/A |
| **Query Body sample** | N/A |
| **Response Body sample** | [  {  "id": *<Old Integration Error Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Error",  "resolved": true  ...  }  ] |

|  |  |
| --- | --- |
| **Step 7a** | Create Integration History (Success)  It must reference the Integration Run |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_history\_records |
| **Query Params** | fields=["id","document-id","status"] |
| **Sample URL** | https://*<your instance hostname>*/api/integration\_history\_records?fields=["id","document-id","status"] |
| **Query Body sample** | {  "document-type": "OrderHeader",  "document-id": *<Purchase Order id>*,  "document-status":"*<Purchase Order status>*",  "contact-alert-type": "Functional",  "status": "**Success**",  "integration": {"code":"*<Customer Integration id>*"},  "integration-run": {"id":*<Integration Run ID>*},  "responses": [  {  "response-code": "Success-1234",  "response-message": "The integration in the ERP went well"  }  ]} |
| **Response Body sample** | {  "id": *<New Integration History Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Success"  } |

|  |  |
| --- | --- |
| **Step 7b** | Create Integration Error and alert to Integration Contact  It must reference the Integration Run |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_errors/create\_alert |
| **Query Params** | N/A |
| **Query Body sample** | {  "document-type": "OrderHeader",  "document-id": *<Purchase Order id>*,  "document-status":"*<Purchase Order status>*",  "contact-alert-type": "Functional",  "status": "**Error**",  "integration-run-id": "*<Integration Run ID>*",  "responses": [  {  "response-code": "Failure-CC",  "response-message": "Line 1: Cost center CA234 is closed for ordering"  },  {  "response-code": "Failure-date",  "response-message": "Line 2: Delivery date cannot be in the past"  }  ]  } |
| **Response Body sample** | {  "id": *<New Integration Error Record id>*,  "document-id": *<Purchase Order id>*,  "status": "Error"  ...  } |

## 

# 

|  |  |
| --- | --- |
| **Step 8** | Successfully End Integration Run |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_runs/*<Integration Run ID>*/success |
| **Query Params** | N/A |
| **Query Body sample** | N/A |
| **Response Body sample** | {  "id": *<Integration Run ID>*,  "status": "successful"  ...  } |

If there is a general failure during the Integration run

|  |  |
| --- | --- |
| **Step x** | Raise Failure for the Integration Run |
| **Method** | PUT |
| **API** | https://*<your instance hostname>*/api/integration\_runs/*<Integration Run ID>*/fail |
| **Query Params** | N/A |
| **Query Body sample** | N/A |
| **Response Body sample** | {  "id": *<Integration Run ID>*,  "status": "failed"  ...  } |