Optimization Configuration

Last Modified on 05/16/2025 8:44 am EDT

V1.1

Overview

The optimization step performs data massaging after the invoice is submitted in the Eyeball task, to ensure the data is clean and ready. To optimize your processing, you can define rules in the Optimization.yaml configuration file.

For example, one such rule can be to copy the content from header level and insert the copied values in the line items. This rule checks whether the UOM (Unit of Measure) used in the data exists in the master data. This rule checks the sum of individual line-item amounts matches the total invoice amount. This rule ensures the PO number is assigned to each invoice line item.

Template

```
kind: ruleSet
metadata:
  name: extraction/v1/documents/optimization
spec:
  nodes:
    # Unit of Measurement (UOM) Validation - Step 1 - fetch master data  UOM collection
  - if: Try(@tempUOM == null)
    then:
      name: tempUOM
      value: ':Try(Join(@Data.Query("UOM").Select(Uom), "; ", true))'
      #value: ':Try(Join(new String[] {"Cs; Kg; Oz"}, "; ", true))'

  # Sum Amount Field
  - if: 'Try( @GST_HST != null && @ecologyDeposit != null)'
    then:
      name: otherCharges
      value: ':Try(ToDecimal(@GST_HST) + ToDecimal(@ecologyDeposit))'

  # Copy PO from header level to line level
  - if: 'Try(@Level == "InvoiceLineItem")'
    nodes:
      - if: 'Try(!string.IsNullOrEmpty(@PurchaseOrder))'
        then:
          name: PurchaseOrder
          value: ':Try(@PurchaseOrder)'

  # Unit of Measurement (UOM) Validation - Step 2 (WIP)
  - if: it["Unit"]?.Value != null && ToString(@tempUOM).length > 0 && !ToString(@tempUOM)?.Contains(ToString(it["
Unit"]?.Value)) && !@RejectReason?.ToString()?.Contains("UOM Mis-match")
    then:
      name: RejectReason
      value: ':Try(Join(new String[] {@RejectReason, "UOM Mis-match"}, "; ", true))'
```

Parameter	Description
If then	The rule/condition based on which optimization is done.