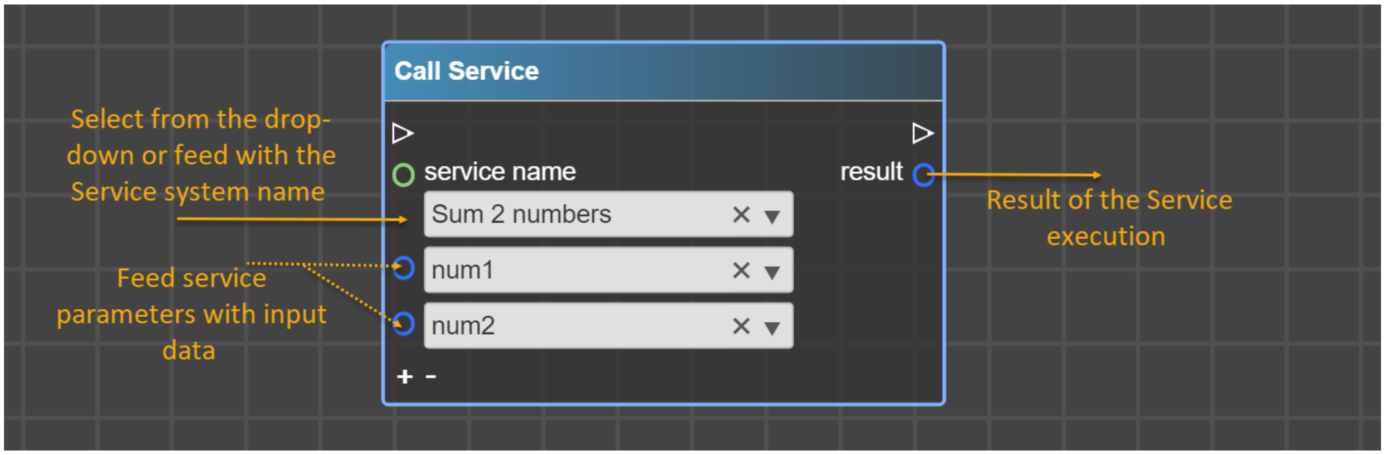
Possibility to call a Flowchart from another Flowchart

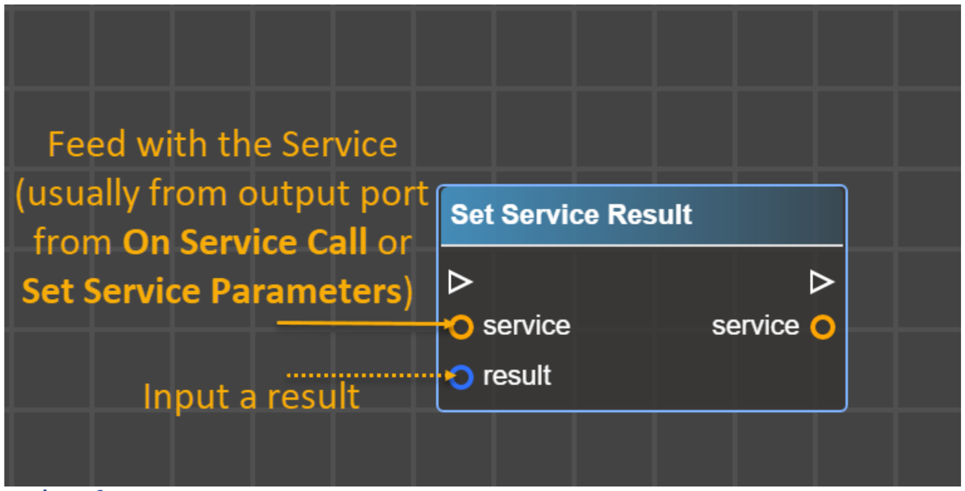
New operator **Call Service** allows calling a flowchart from the other flowchart:


Call Service
service name
result
Sum 2 numbers
num1
num2
+ -
Select from the drop-down or feed with the Service system name
Feed service parameters with input data
Result of the Service execution

Select Service from the dropdown and provide Service Parameters with data. The flowchart implementing this service will be called and provide the result which can be used further in this flowchart.

Prior to calling the Service, please perform the following steps:

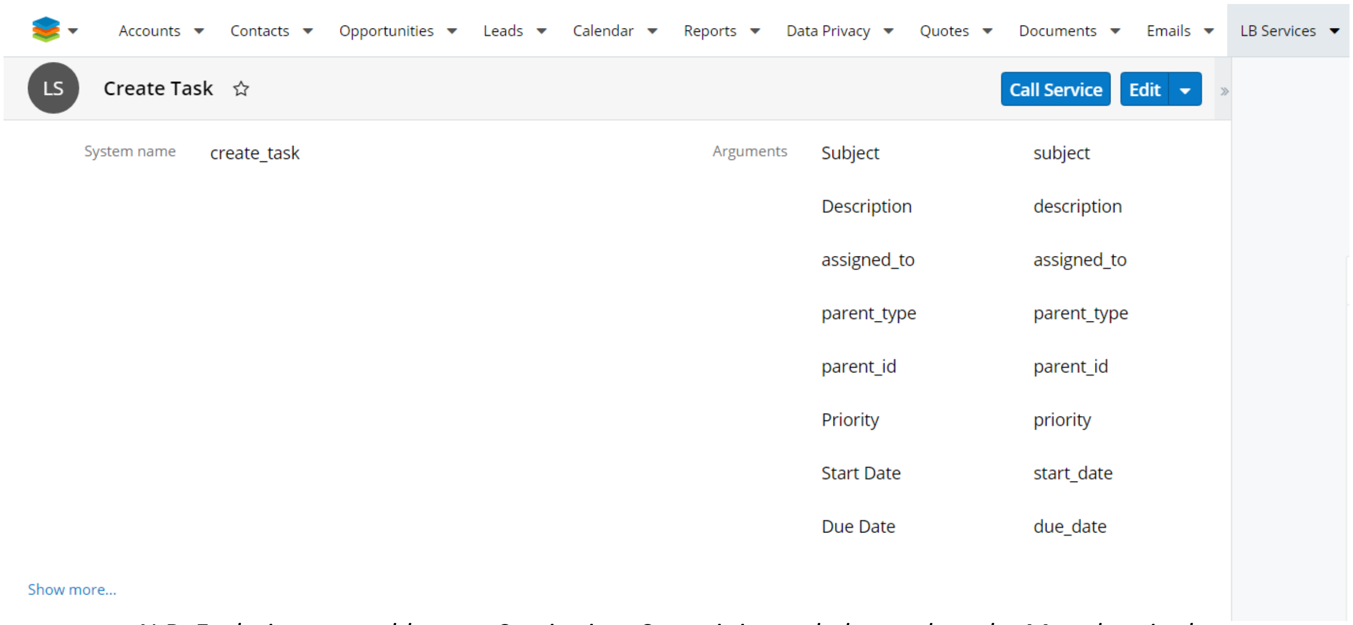
1. **LB Services** Extension must be deployed on Sugar Instance. Find and download here LB Services Extension version that corresponds to your Sugar version.
2. Service must be registered in Sugar - it must be a record in LB Service module with specified *System Name* and service *Arguments*.
3. Flowchart that implements Service must be deployed.
   Such a flowchart starts with **On Service Call** and describes what should happen when Service is called. When a result is expected (data in *result* port in Call Service operator), this result must be set with operator **Set Service Result** in the flowchart that implements Service:


Set Service Result
service
service
result
Feed with the Service (usually from output port from On Service Call or Set Service Parameters)
Input a result

Example of usage

Let's say there are a bunch of flowcharts with the same steps to create a Task. Instead of performing the same steps in each flowchart, it is possible to create the Service and then re-use it.

1. Register Service in Sugar.
For this, just create a record in LB Services module and provide arguments that are needed to be filled to create Task:

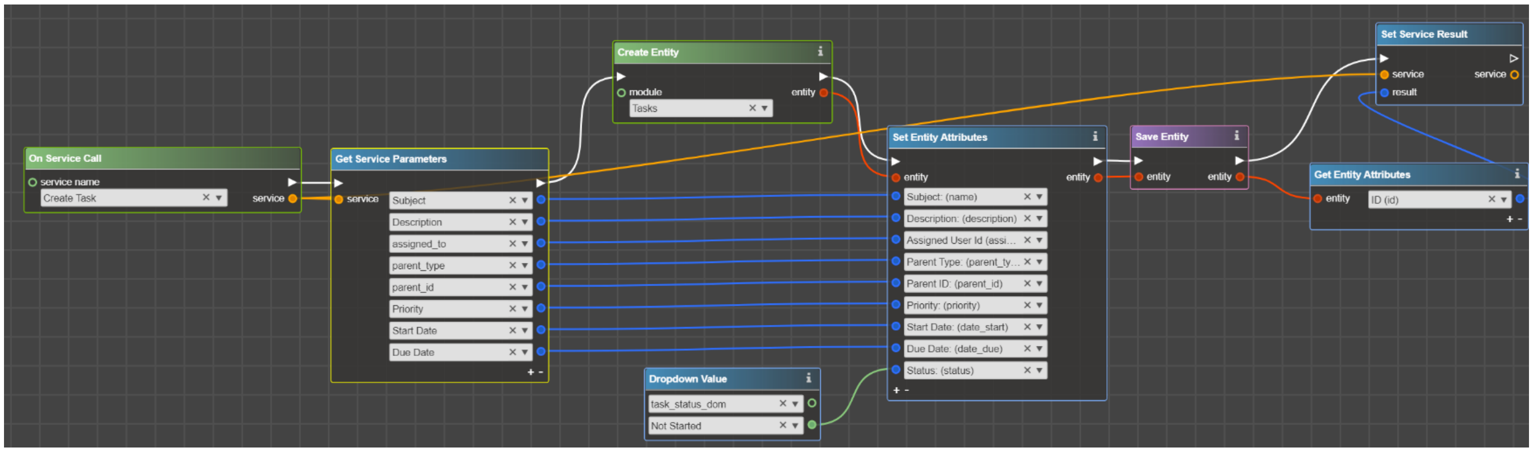


System name create_task

Arguments	
Subject	subject
Description	description
assigned_to	assigned_to
parent_type	parent_type
parent_id	parent_id
Priority	priority
Start Date	start_date
Due Date	due_date

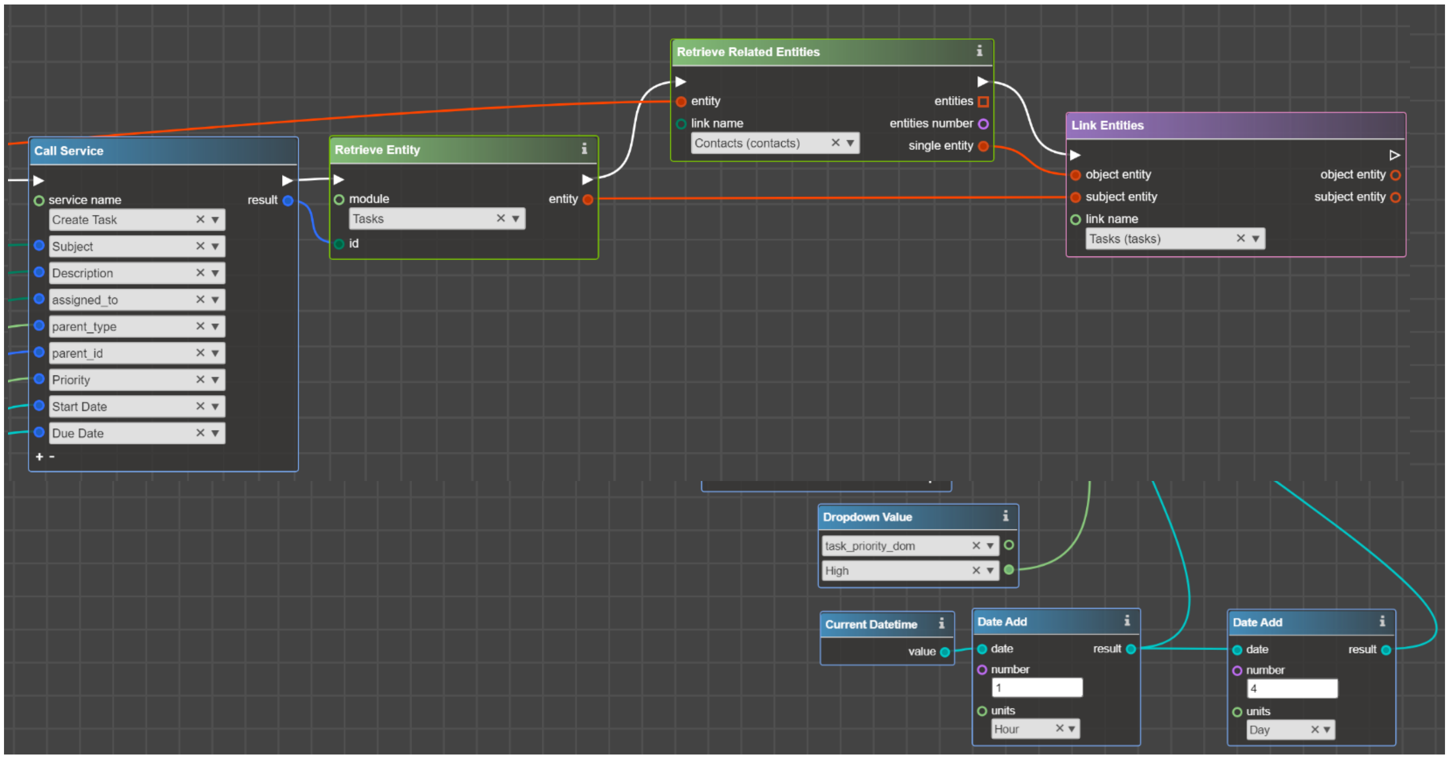
N.B. Each time you add a new Service into Sugar it is needed to update the Metadata in the Logic Builder to work with the newly added Service.

2. Crete a Flowchart starting from **On Service Call** to create a Task using parameters that will be input in the Service. In a *result* input the Task ID.



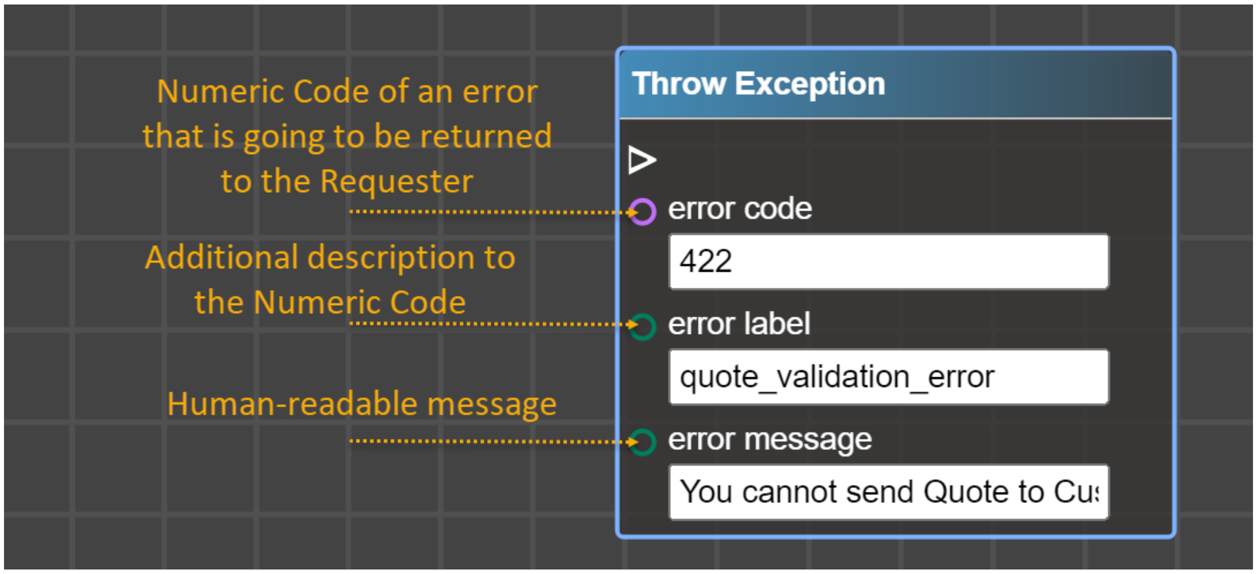
3. Deploy the flowchart into Sugar. Now you can use created Service in other flowcharts, e.g when Case with Type Product is added:

Also, it is possible to link the Task to other entities using *result* output port from the **Call Service**:



New Validation Possibilities

Operator **Throw Exception** allows addressing the challenges of data validation. For example, when data comes through API to Sugar from an external system – prior to applying changes or saving a record you could implement additional rules, and return an error when rules are violated.
Throw Exception result is also seen in interface to users.


Throw Exception
Numeric Code of an error that is going to be returned to the Requester
Additional description to the Numeric Code
Human-readable message

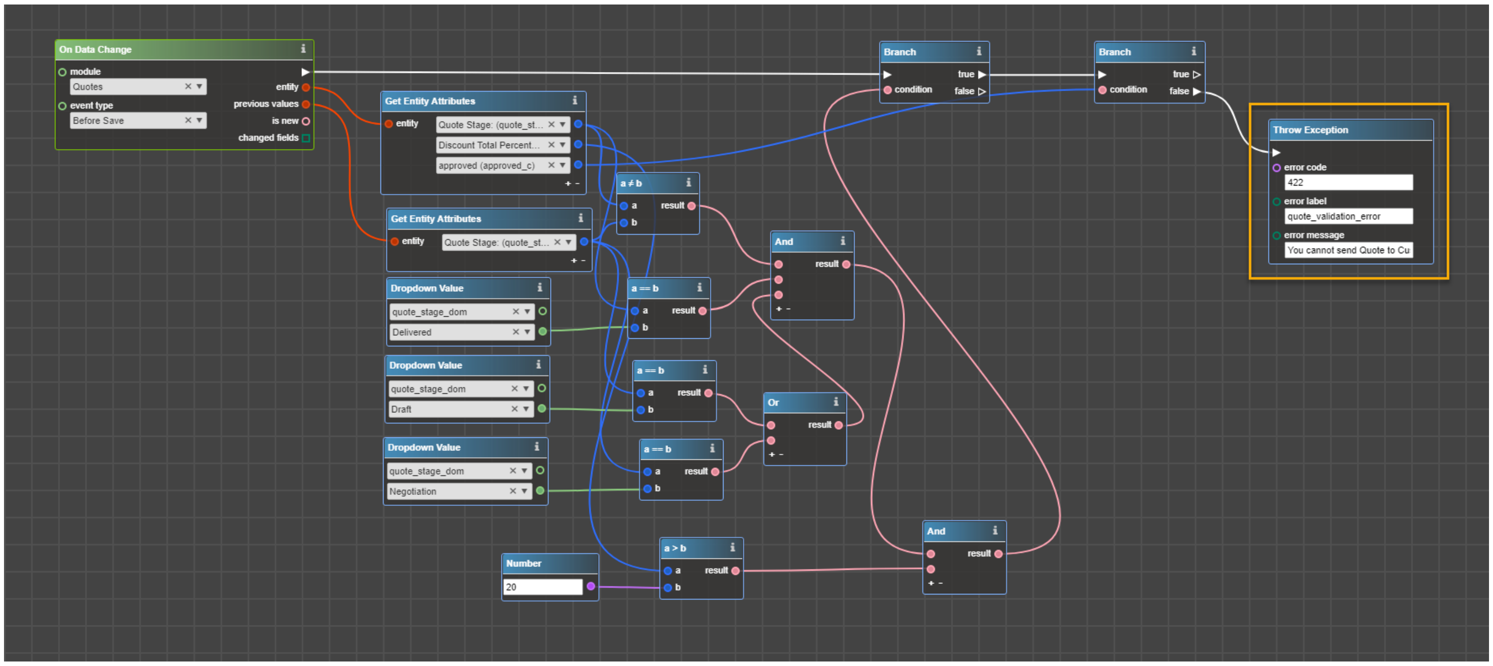
error code
422
error label
quote_validation_error
error message
You cannot send Quote to Cu

N.B. For validation rules, use error code 422

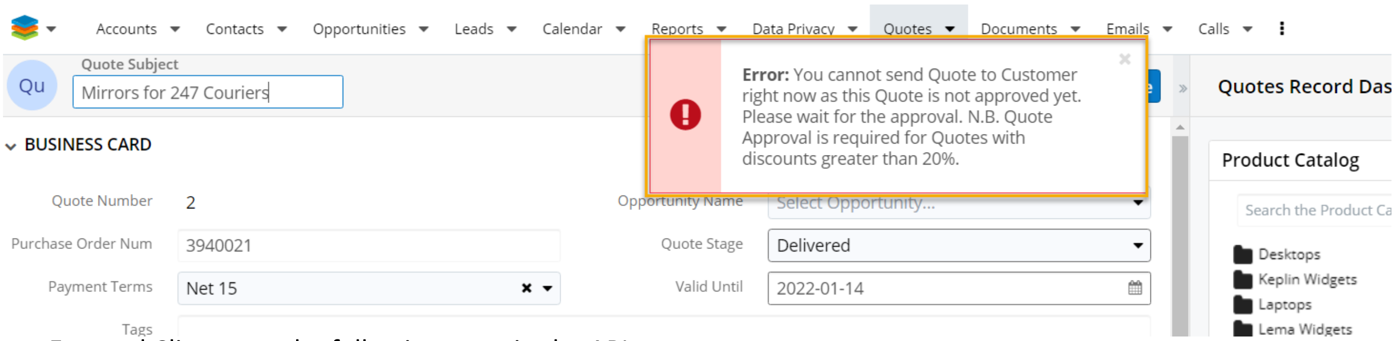
Example of usage

Let’s say that Quotes with “Discount Amount” more than 20% must be approved by Manager. To implement this rule, a Quote must not be moved forward prior to the approval either by a user or by API request.

So the flowchart which implements all sets of checks ends with Throw Exception when conditions aren’t met:



As a result, when criteria of the rule is violated, a user sees the following error in the interface:

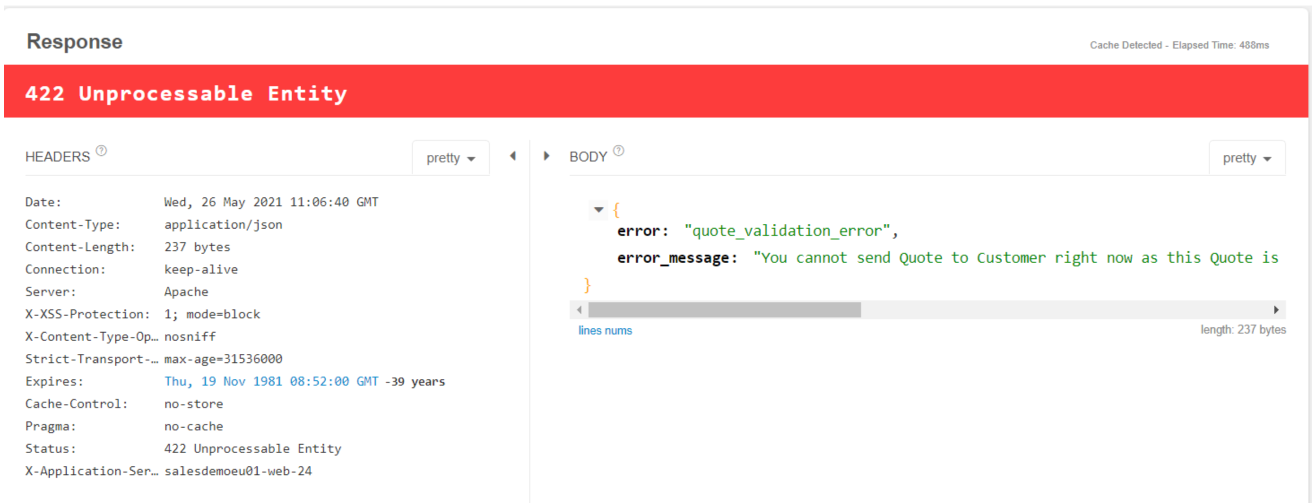


Quote Subject: Mirrors for 247 Couriers

Error: You cannot send Quote to Customer right now as this Quote is not approved yet. Please wait for the approval. N.B. Quote Approval is required for Quotes with discounts greater than 20%.

Quote Number	2	Opportunity Name	Select Opportunity...
Purchase Order Num	3940021	Quote Stage	Delivered
Payment Terms	Net 15	Valid Until	2022-01-14
Tags			

External Clients get the following error in the API response:



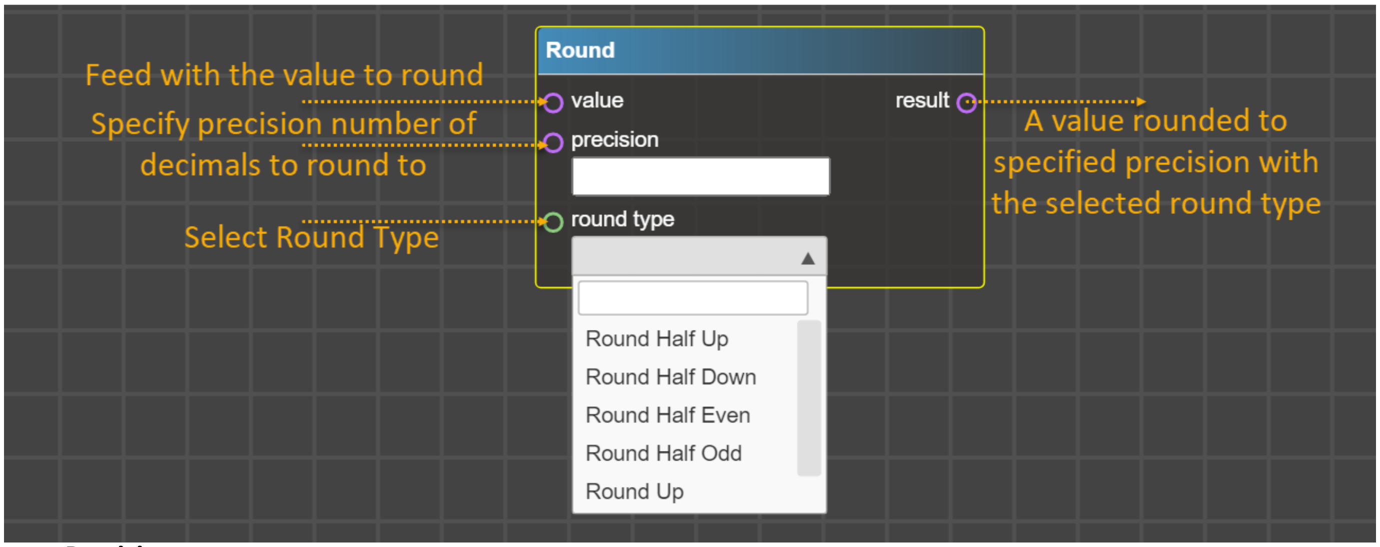
422 Unprocessable Entity

```
Date:                Wed, 26 May 2021 11:06:40 GMT
Content-Type:        application/json
Content-Length:      237 bytes
Connection:          keep-alive
Server:              Apache
X-XSS-Protection:    1; mode=block
X-Content-Type-Op…   nosniff
Strict-Transport-…   max-age=31536000
Expires:             Thu, 19 Nov 1981 08:52:00 GMT -39 years
Cache-Control:       no-store
Pragma:              no-cache
Status:              422 Unprocessable Entity
X-Application-Ser…   salesdemoeu01-web-24
```

```
{
  error:  "quote_validation_error",
  error_message:  "You cannot send Quote to Customer right now as this Quote is
}
```

Possibility to round numbers

Operator **Round** allows rounding a value to specified precision (number of digits after the decimal point):


Feed with the value to round
Specify precision number of decimals to round to
Select Round Type
Round
value
precision
round type
result
A value rounded to specified precision with the selected round type
Round Half Up
Round Half Down
Round Half Even
Round Half Odd
Round Up

Precision:

- If the precision is positive, a value is rounded to defined in precision significant digits after the decimal point.
- If the precision is negative, a value is rounded to defined in precision significant digits before the decimal point (e.g. for a precision of -1 value is rounded to tens, for a precision of -2 to hundreds, etc.)

Round Type:

- *Round Half Up* Rounds a value away from zero when it is half way there, making 1.5 into 2 and -1.5 into -2.
- *Round Half Down* Rounds a value towards zero when it is half way there, making 1.5 into 1 and -1.5 into -1.
- *Round Half Even* Rounds a value towards the nearest even value when it is half way there, making both 1.5 and 2.5 into 2.
- *Round Half Odd* Rounds a value towards the nearest odd value when it is half way there, making 1.5 into 1 and 2.5 into 3.
- *Round Up* Rounds a value away from zero, making 1.5 into 2
- *Round Down* Rounds a value towards zero, making 1.5 into 1

Sugar 11.0.x Compatibility

Packages generated from Logic Builder are fully compatible with Sugar 11.0.x (Q2 2021 release).