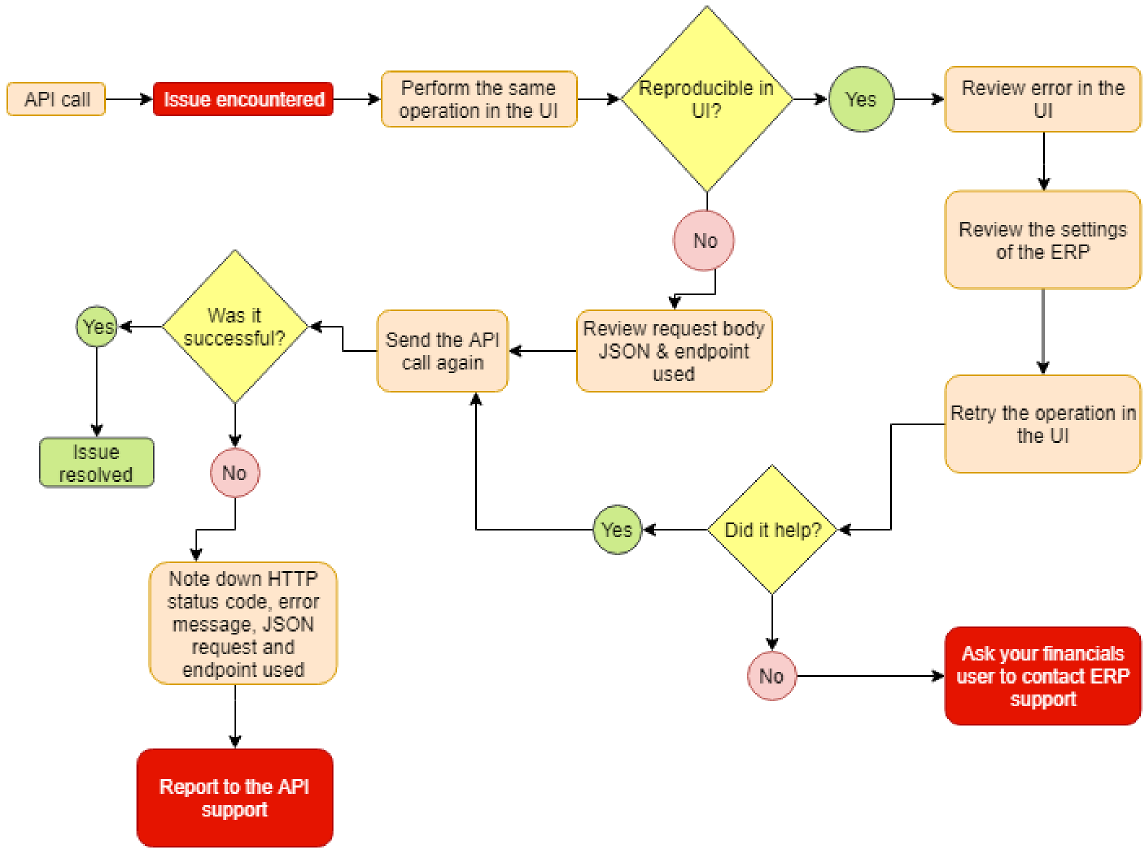
API incidents - troubleshooting and reporting

Encountering an Issue

First troubleshooting steps

When encountering an issue using the API there are steps you'll want to follow to understand where the issue occurs and what support channels it should be reported via to resolve the issue in a timely manner.

1. Review the error message, look at the returned HTTP status code to understand what the issue is regarding.
2. Perform the same action in the UI to determine if it is API specific or if it is the UI that is affected.
    a. If the issue is reproducible in the UI, review the error message in the UI, this will often point you to where in the ERP you might need to review settings. If this doesn't help, inform your ERP UI user that they need to open a case through the ERP support channels specific to the market.
    b. If the issue is not reproducible in the UI, review the action being performed via the API.
        i. Are you using the correct endpoint for what you want to do?
        ii. Can the error message inform you what is going wrong? You might need to review the JSON request body, to see the syntax and required fields for the endpoint, please have a look at Visma.Net API Swagger.
        iii. If you have reviewed this and you are still unable to resolve the issue, head over to the Developer Forum and search for similar issues or send an email to developersupport@visma.com
        iv. If you are still unable to find a solution, create a forum thread and include all relevant information, further explained later in this document, so that the API support team may investigate your issue.


API call
Issue encountered
Perform the same operation in the UI
Reproducible in UI?
Yes
Review error in the UI
Review the settings of the ERP
Retry the operation in the UI
Did it help?
Yes
No
Ask your financials user to contact ERP support
No
Review request body JSON & endpoint used
Send the API call again
Was it successful?
Yes
Issue resolved
No
Note down HTTP status code, error message, JSON request and endpoint used
Report to the API support

Reporting the issue

The terms for the API support states that to make investigation easier and reduce the time required to find a solution to your problem, you'll need to include as much relevant information as possible when you are reporting your issue, this is true regardless of what channel your issue should be reported to.

When reporting an issue to the developer forum the thread should contain the following information and be formatted in a way that makes the post to be easy to follow.

What are you trying to do

Explain what you are trying to do, what the expected results are and what results you get.

Provide detailed steps of the work flow you have followed.

Your Request:

1. Method and URL to endpoint where issue occurs
   a. e.g. **GET https://integration.visma.net/API/controller/api/v1/inventory/**
2. JSON request body
   a. Post this in code blocks: **[CODE]** { "My JSON" : { "Value": "Request" } } **[/CODE]**

The servers Response:

1. HTTP Status Code

    a. Post this in code blocks: **[CODE]** 201 Created **[/CODE]**
2. JSON response body
    a. Post this in code blocks: **[CODE]** { “Servers JSON” : { “Value”: “Response” } } **[/CODE]**

Case Example

Here are two examples of simple cases and the steps you might take.

Swagger - How to troubleshoot using the documentation

Often issues encountered via the API are related to the syntax of the request you are sending, and this is also where you should first try to catch any errors. Below is an example of when checking the documentation for the operation/endpoint is the solution to your issue.

Example case 1: POST SalesOrder - "Operation parameter: value 0 - 0 is not valid for this type of request."

When posting a SalesOrder I’ve encountered a problem. I am sure I am using the correct endpoint and method, but I have not yet checked the syntax of the operation.

Endpoint and method:

```
POST: https://integration.visma.net/API/controller/api/v1/salesorder
```

Request JSON body:

```
{
   "customer": {
      "value": "10000"
   },
   "orderNumber":{
            "value":"7777"
   },
   "orderType": {
      "value": "SO"
   },
   "lines": [
      {
         "unitPrice": {
            "value": 1131.25
         },
         "quantity": {
            "value": 1
         },
         "inventoryId": {
            "value": "6"
         },
         "warehouse": {
            "value": "2"
         },
         "lineDescription": {
            "value": "Betaling"
         }
      }
   ],
   "currency": {
      "value": "NOK"
   },
   "date": {
      "value": "2019-06-19T13:57:42Z"
   }
}
```

HTTP Status Code:

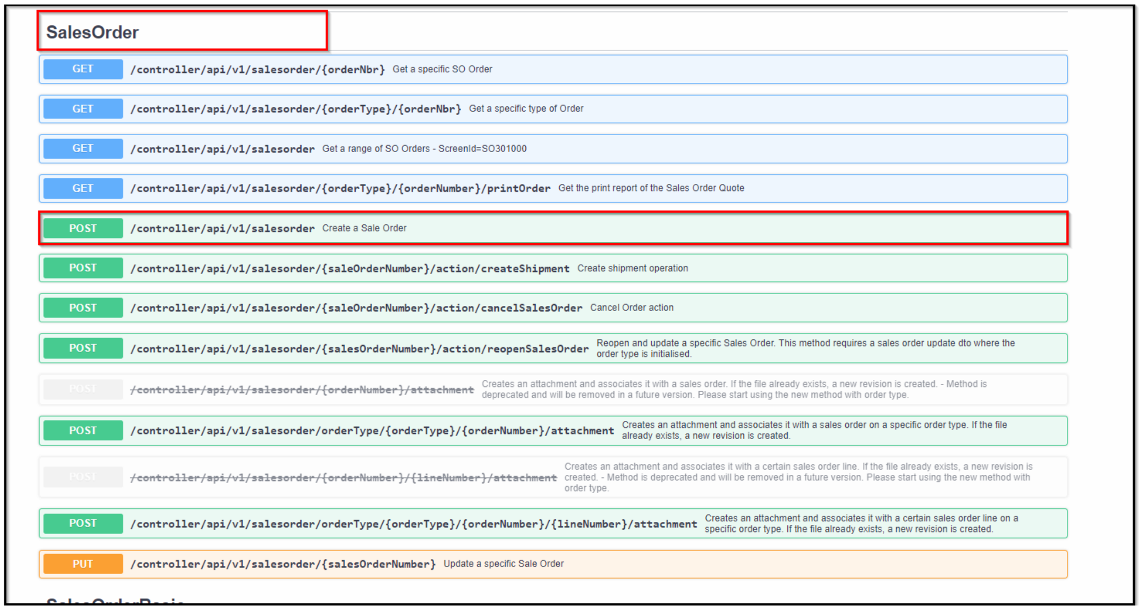
```
400 Bad Request
```

Response Body - JSON:

```
{
  "message": "Operation parameter: value 0 - 0 is not valid for this type of request."
}
```

Documentation for the endpoint in Swagger

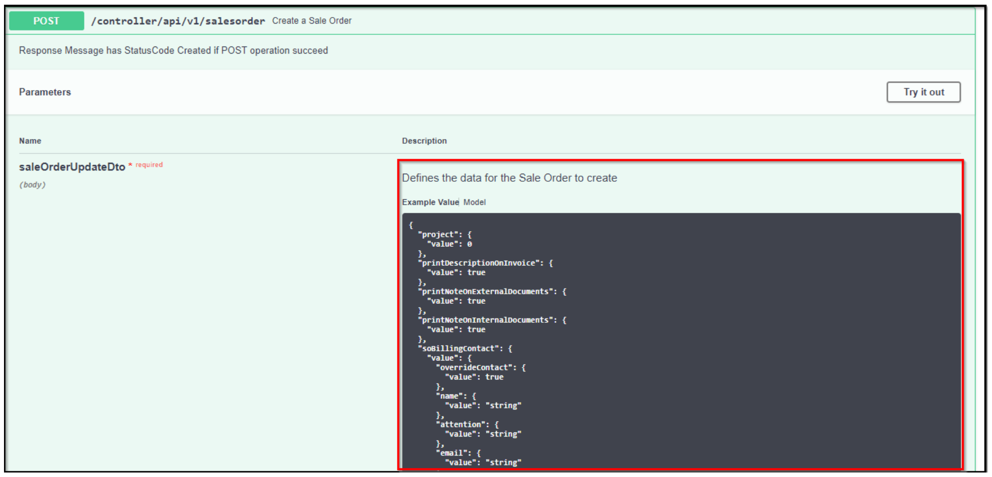
First, scroll down to the endpoint and expand it:



SalesOrder

- GET `/controller/api/v1/salesorder/{orderNbr}` Get a specific SO Order
- GET `/controller/api/v1/salesorder/{orderType}/{orderNbr}` Get a specific type of Order
- GET `/controller/api/v1/salesorder` Get a range of SO Orders - ScreenId=SO301000
- GET `/controller/api/v1/salesorder/{orderType}/{orderNumber}/printOrder` Get the print report of the Sales Order Quote
- POST `/controller/api/v1/salesorder` Create a Sale Order
- POST `/controller/api/v1/salesorder/{saleOrderNumber}/action/createShipment` Create shipment operation
- POST `/controller/api/v1/salesorder/{saleOrderNumber}/action/cancelSalesOrder` Cancel Order action
- POST `/controller/api/v1/salesorder/{salesOrderNumber}/action/reopenSalesOrder` Reopen and update a specific Sales Order. This method requires a sales order update dto where the order type is initialised.
- POST ~~`/controller/api/v1/salesorder/{orderNumber}/attachment`~~ Creates an attachment and associates it with a sales order. If the file already exists, a new revision is created. - Method is deprecated and will be removed in a future version. Please start using the new method with order type.
- POST `/controller/api/v1/salesorder/orderType/{orderType}/{orderNumber}/attachment` Creates an attachment and associates it with a sales order on a specific order type. If the file already exists, a new revision is created.
- POST ~~`/controller/api/v1/salesorder/{orderNumber}/{lineNumber}/attachment`~~ Creates an attachment and associates it with a certain sales order line. If the file already exists, a new revision is created. - Method is deprecated and will be removed in a future version. Please start using the new method with order type.
- POST `/controller/api/v1/salesorder/orderType/{orderType}/{orderNumber}/{lineNumber}/attachment` Creates an attachment and associates it with a certain sales order line on a specific order type. If the file already exists, a new revision is created.
- PUT `/controller/api/v1/salesorder/{salesOrderNumber}` Update a specific Sale Order

Expand the method:



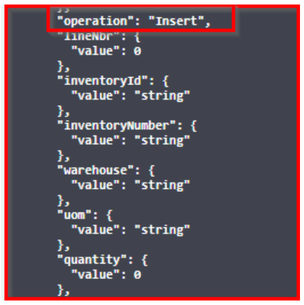
POST `/controller/api/v1/salesorder` Create a Sale Order

Response Message has StatusCode Created if POST operation succeed

Name	Description
saleOrderUpdateDto * required (body)	Defines the data for the Sale Order to create

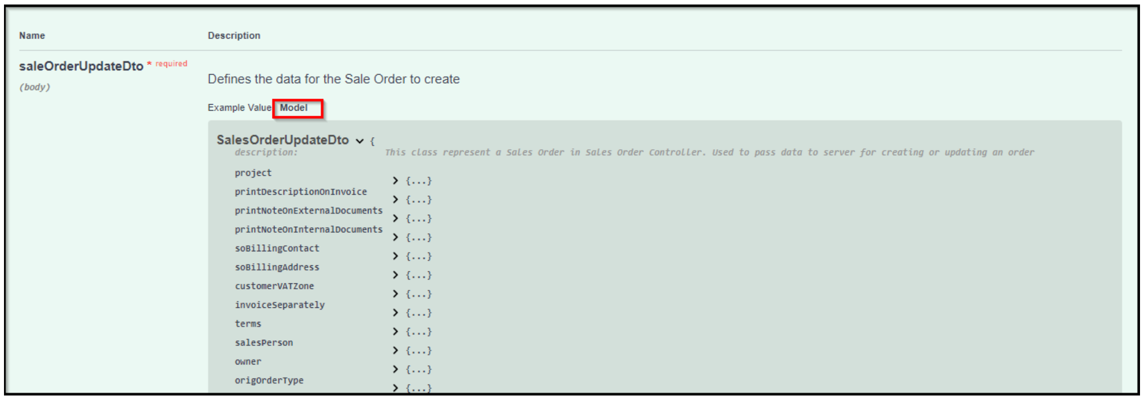
```
{
  "project": {
    "value": 0
  },
  "printDescriptionOnInvoice": {
    "value": true
  },
  "printNoteOnExternalDocuments": {
    "value": true
  },
  "printNoteOnInternalDocuments": {
    "value": true
  },
  "soBillingContact": {
    "value": {
      "overrideContact": {
        "value": true
      },
      "name": {
        "value": "string"
      },
      "attention": {
        "value": "string"
      },
      "email": {
        "value": "string"
```

In this example, you'll see that for the lines, you need to specify an operation to perform:



```
  "operation": "Insert",
  "lineNbr": {
    "value": 0
  },
  "inventoryId": {
    "value": "string"
  },
  "inventoryNumber": {
    "value": "string"
  },
  "warehouse": {
    "value": "string"
  },
  "uom": {
    "value": "string"
  },
  "quantity": {
    "value": 0
  },
```

By clicking "model" over the example, you can check what each values each key accepts:



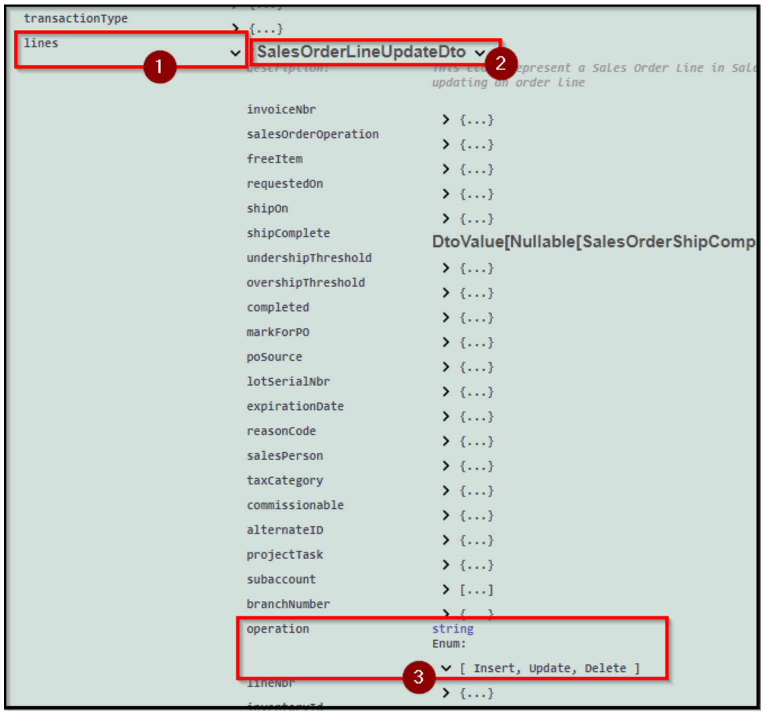
Name	Description
saleOrderUpdateDto * required (body)	Defines the data for the Sale Order to create

Example Value Model

```
SalesOrderUpdateDto {
  description:                This class represent a Sales Order in Sales Order Controller. Used to pass data to server for creating or updating an order
  project                     > {...}
  printDescriptionOnInvoice   > {...}
  printNoteOnExternalDocuments > {...}
  printNoteOnInternalDocuments > {...}
  soBillingContact            > {...}
  soBillingAddress            > {...}
  customerVATZone             > {...}
  invoiceSeparately           > {...}
  terms                       > {...}
  salesPerson                 > {...}
  owner                       > {...}
  origOrderType               > {...}
```

Expanding the relevant line until you see the keys will show you what is accepted:

1. Expand "lines"
2. Expand "SalesOrderLineUpdateDto"
3. Expand "operation: String, Enum:"



```
transactionType          > {...}
lines                    v SalesOrderLineUpdateDto v
                           description:          This class represent a Sales Order Line in Sale
                                                 updating an order line
                           invoiceNbr            > {...}
                           salesOrderOperation   > {...}
                           freeItem              > {...}
                           requestedOn           > {...}
                           shipOn                > {...}
                           shipComplete          DtoValue[Nullable[SalesOrderShipComp
                           undershipThreshold    > {...}
                           overshipThreshold     > {...}
                           completed             > {...}
                           markForPO             > {...}
                           poSource              > {...}
                           lotSerialNbr          > {...}
                           expirationDate        > {...}
                           reasonCode            > {...}
                           salesPerson           > {...}
                           taxCategory           > {...}
                           commissionable        > {...}
                           alternateID           > {...}
                           projectTask           > {...}
                           subaccount            > [...]
                           branchNumber          > {...}
                           operation             string
                                                 Enum:
                                                 v [ Insert, Update, Delete ]
                           lineNbr               > {...}
```

This shows you that you have three options to add here: "Insert", "Update" or "Delete".

So in this example each line should look like this:

```
"lines": [
      {
        "operation": "Insert",
         "unitPrice": {
            "value": 1131.25
         },
         "quantity": {
            "value": 1
         },
         "inventoryId": {
            "value": "6"
         },
         "warehouse": {
            "value": "2"
         },
         "lineDescription": {
            "value": "Betaling"
         }
      }
   ]
```

UI - How to troubleshoot using the ERP UI

In the cases where you have checked the documentation for the endpoint you are using the next step would be to troubleshoot the scenario in the UI. Below is an example of when it would be appropriate to head over to the UI to see what happens in the ERP.

Example case 2: POST SalesOrder - “Unable to generate the next number. Manual numbering is activated for 'SOORDER'."

When posting a SalesOrder I’ve encountered a problem. I’ve checked the Swagger documentation and my request JSON follows the requirement, I’m using the correct method and endpoint.

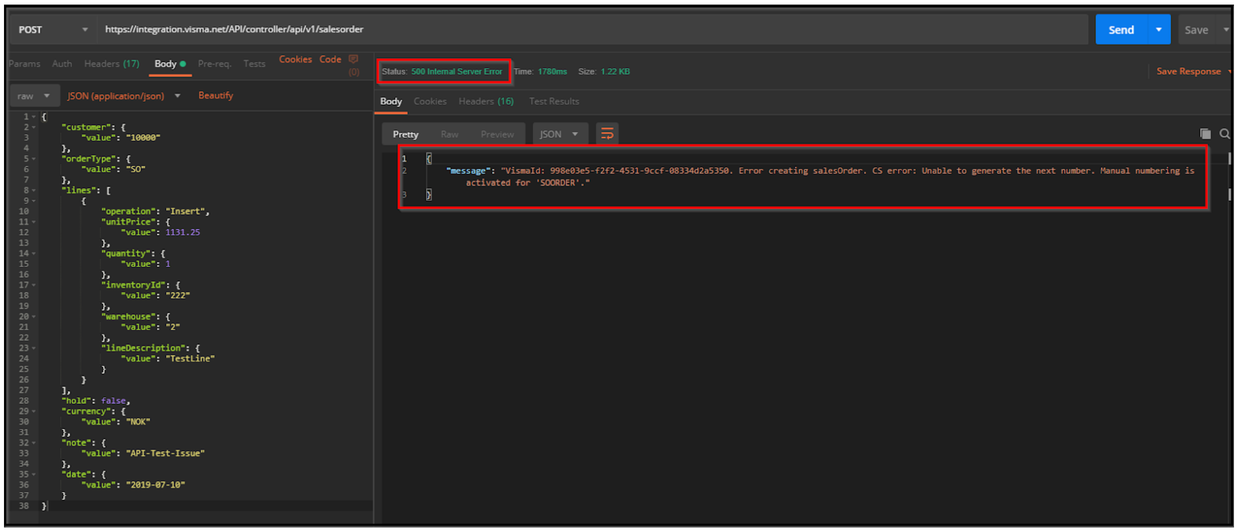
Even though I’ve followed the steps required to successfully post via the API I still get a response with HTTP status 500 and an error message.

HTTP Status Code:

```
500 Internal Server Error
```

Response Body - JSON :

```
{  "message": "VismaId: 4ba5df28-3892-4295-a0eb-fbc668882171. Error creating salesOrder. CS error: Unable to generate the next number. Manual numbering is activated for 'SOORDER'." }
```


POST https://integration.visma.net/API/controller/api/v1/salesorder

```
{
    "customer": {
        "value": "10000"
    },
    "orderType": {
        "value": "SO"
    },
    "lines": [
        {
            "operation": "Insert",
            "unitPrice": {
                "value": 1131.25
            },
            "quantity": {
                "value": 1
            },
            "inventoryId": {
                "value": "222"
            },
            "warehouse": {
                "value": "2"
            },
            "lineDescription": {
                "value": "TestLine"
            }
        }
    ],
    "hold": false,
    "currency": {
        "value": "NOK"
    },
    "note": {
        "value": "API-Test-Issue"
    },
    "date": {
        "value": "2019-07-10"
    }
}
```

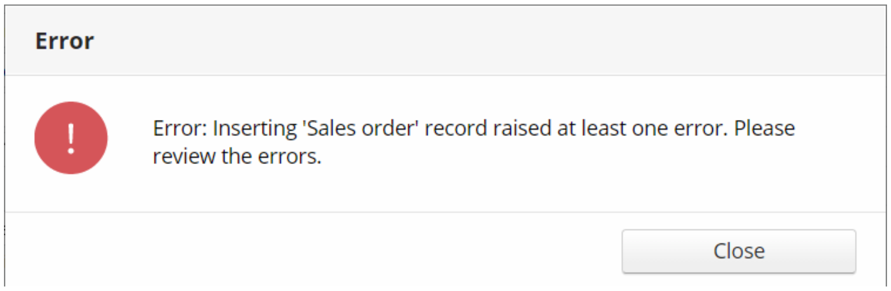
Status: 500 Internal Server Error Time: 1780ms Size: 1.22 KB

```
{
    "message": "VismaId: 998e03e5-f2f2-4531-9ccf-08334d2a5350. Error creating salesOrder. CS error: Unable to generate the next number. Manual numbering is
        activated for 'SOORDER'."
}
```

Test the operation in the UI

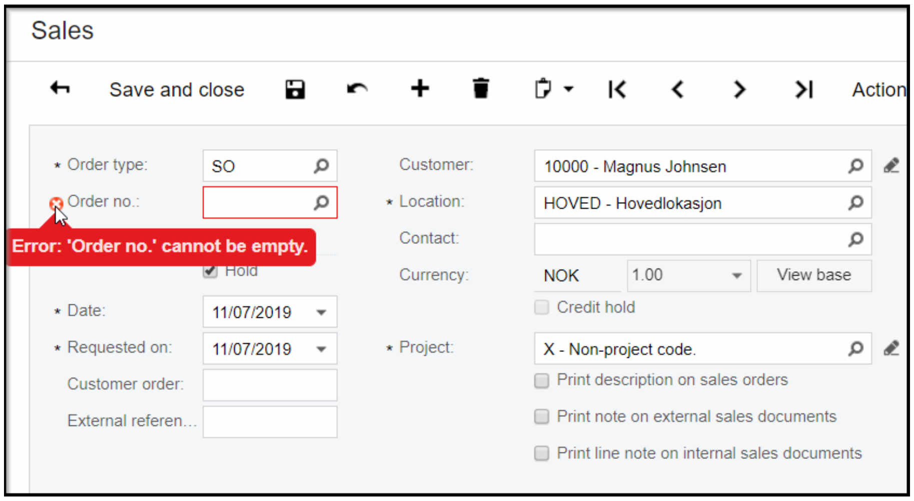
To determine where the issue occurs, I head over to my Visma.Net Financials company to perform the same action in the UI.

I create a new SalesOrder(ScreenId=SO301000), I choose my customer, add a line and save.


Error

Error: Inserting 'Sales order' record raised at least one error. Please review the errors.

I also get an error in the UI, I close the message and review the error.


Sales

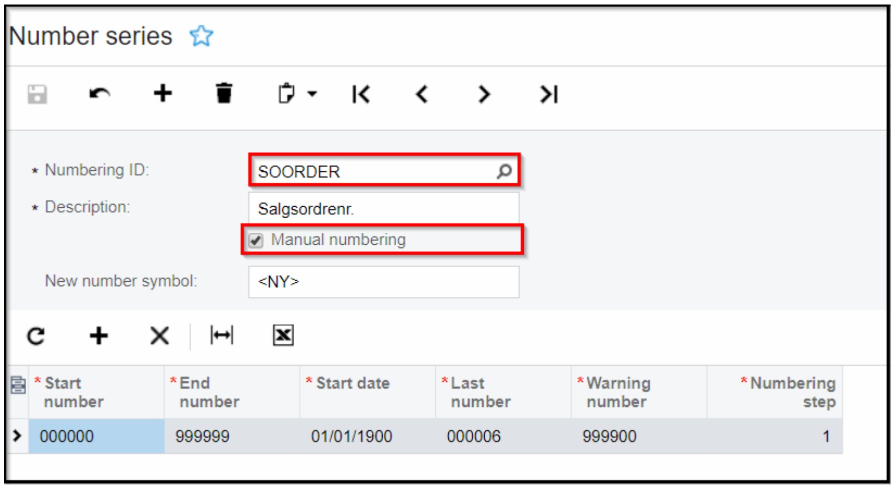
* Order type:	SO	Customer:	10000 - Magnus Johnsen
Order no.:		* Location:	HOVED - Hovedlokasjon
		Contact:	
	Hold	Currency:	NOK 1.00 View base
			Credit hold
* Date:	11/07/2019	* Project:	X - Non-project code.
* Requested on:	11/07/2019		Print description on sales orders
Customer order:			Print note on external sales documents
External referen...			Print line note on internal sales documents

Error: 'Order no.' cannot be empty.

I cannot save the Sales Order without a number. Since I cannot perform the same operation in the UI, the issue does not lie with the API but with the functionality of the ERP system.

Together with the error message I received from the API, I can conclude that the number series for Sales Order requires manual numbering.

When I look at the settings for “Number Series”(ScreenId=CS201010) I see that indeed “Manual numbering” is ticked off.



* Numbering ID:	SOORDER
* Description:	Salgsordrenr.
✓ Manual numbering	
New number symbol:	<NY>

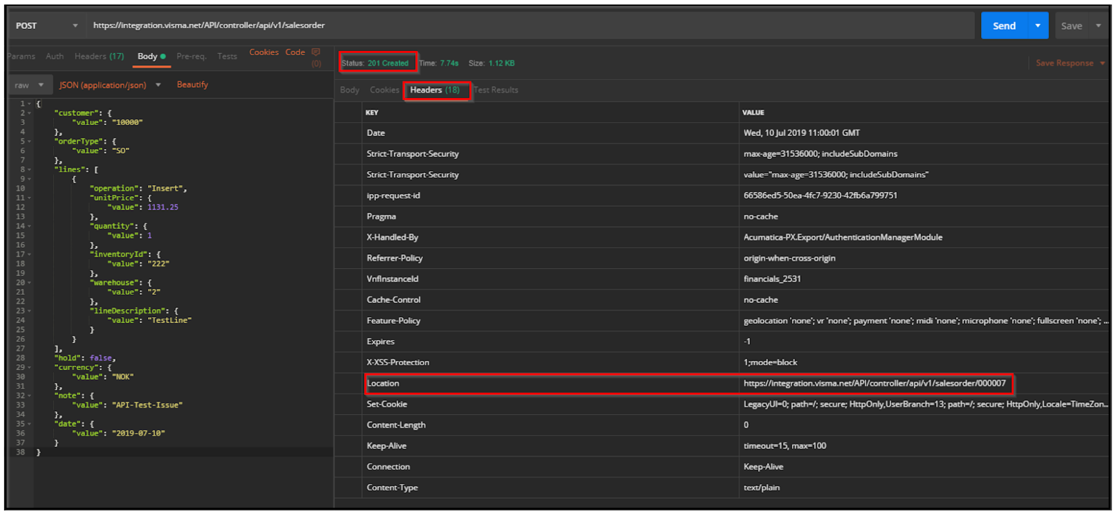
* Start number	* End number	* Start date	* Last number	* Warning number	* Numbering step
000000	999999	01/01/1900	000006	999900	1

From this point I have two options, either I enter the number manually, or I turn of “Manual numbering” for Sales Order.

Resolving the issue

Option 1: Turn of manual numbering

1. If I untick “Manual numbering” I will be able to send the same **JSON** I previously sent without receiving an error.
2. To find the order number created I need to have a look at the **response headers**, under location I’ll find the newly created order in the end of the **location URL**



POST https://integration.visma.net/API/controller/api/v1/salesorder

JSON (application/json)

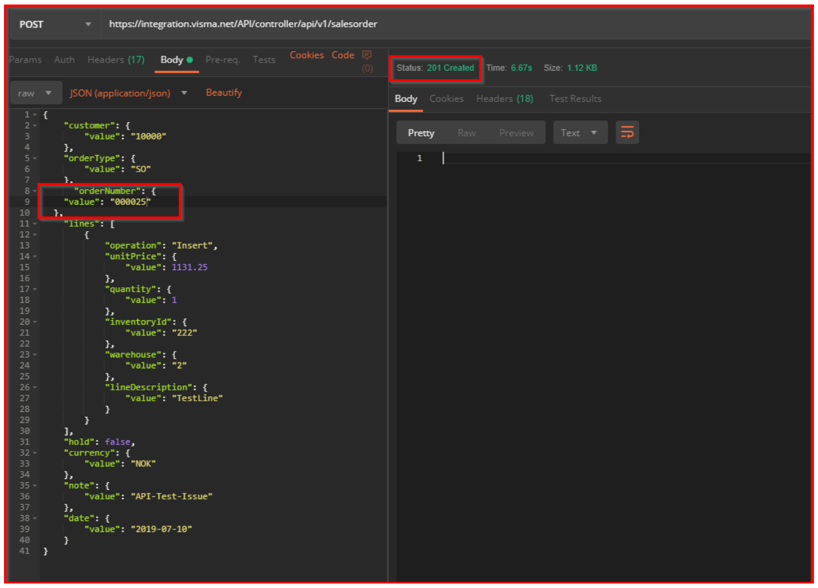
```
{
    "customer": {
        "value": "10000"
    },
    "orderType": {
        "value": "SO"
    },
    "lines": [
        {
            "operation": "Insert",
            "unitPrice": {
                "value": 1131.25
            },
            "quantity": {
                "value": 1
            },
            "inventoryId": {
                "value": "222"
            },
            "warehouse": {
                "value": "2"
            },
            "lineDescription": {
                "value": "TestLine"
            }
        }
    ],
    "hold": false,
    "currency": {
        "value": "NOK"
    },
    "note": {
        "value": "API-Test-Issue"
    },
    "date": {
        "value": "2019-07-10"
    }
}
```

Status: 201 Created Time: 7.74s Size: 1.12 KB

KEY	VALUE
Date	Wed, 10 Jul 2019 11:00:01 GMT
Strict-Transport-Security	max-age=31536000; includeSubDomains
Strict-Transport-Security	value="max-age=31536000; includeSubDomains"
ipp-request-id	66586ed5-50ea-4fc7-9230-42fb6a799751
Pragma	no-cache
X-Handled-By	Acumatica-PX.Export/AuthenticationManagerModule
Referrer-Policy	origin-when-cross-origin
VnfInstanceId	financials_2531
Cache-Control	no-cache
Feature-Policy	geolocation 'none'; vr 'none'; payment 'none'; midi 'none'; microphone 'none'; fullscreen 'none'; ...
Expires	-1
X-XSS-Protection	1;mode=block
Location	https://integration.visma.net/API/controller/api/v1/salesorder/000007
Set-Cookie	LegacyUI=0; path=/; secure; HttpOnly,UserBranch=13; path=/; secure; HttpOnly,Locale=TimeZon...
Content-Length	0
Keep-Alive	timeout=15, max=100
Connection	Keep-Alive
Content-Type	text/plain

Option 2: Sending the order number

1. If I still want the numbering to be manual I'll have to go back to my request and insert the line for "orderNumber"
2. When I do this, I get the HTTP status code response: 201 Created, letting me know the operation was successful.


POST
https://integration.visma.net/API/controller/api/v1/salesorder
Params Auth Headers (17) Body Pre-req. Tests Cookies Code (0)
raw JSON (application/json) Beautify
Status: 201 Created Time: 6.67s Size: 1.12 KB
Body Cookies Headers (18) Test Results
Pretty Raw Preview Text

```
{
    "customer": {
        "value": "10000"
    },
    "orderType": {
        "value": "SO"
    },
      "orderNumber": {
    "value": "000025"
  },
    "lines": [
        {
            "operation": "Insert",
            "unitPrice": {
                "value": 1131.25
            },
            "quantity": {
                "value": 1
            },
            "inventoryId": {
                "value": "222"
            },
            "warehouse": {
                "value": "2"
            },
            "lineDescription": {
                "value": "TestLine"
            }
        }
    ],
    "hold": false,
    "currency": {
        "value": "NOK"
    },
    "note": {
        "value": "API-Test-Issue"
    },
    "date": {
        "value": "2019-07-10"
    }
}
```

Reporting an unresolved issue

If I would have been unable to resolve the above issue after taking the steps necessary to troubleshoot the issue and I have determined that the operation is successful in the UI, but when it is performed via the API, it doesn't go through, I'll create a thread on the forum.

Here I detail:

- The steps I have taken
- The method I'm using
- The endpoint I'm contacting,
- I include my JSON request body,
- The servers HTTP status code
- The response body

My thread;

Hi!

I'm trying to create a SalesOrder, but I ran into a problem and get an error message back.

I've tried to perform the same action in the UI with success.

This is what I've done:

POST https://integration.visma.net/API/controller/api/v1/salesorder

JSON:

```
[CODE]{
   "customer": { "value": "10000"  },
   "orderType": { "value": "SO"  },
   "lines": [
      {  "operation": "Insert",
         "unitPrice": { "value": 1131.25 },
         "quantity": { "value": 1 },
         "inventoryId": { "value": "222" },
         "warehouse": { "value": "2" },
         "lineDescription": { "value": "TestLine" } } ],
   "hold": false,
   "currency": { "value": "NOK" },
   "note": { "value": "API-Test-Issue"  },
   "date": {"value": "2019-07-10" }}
[/CODE]
```

I get the following response:

HTTP Status Code: 500 Internal Server Error

Response body - JSON:

```
[CODE] {  "message": "VismaId: 4ba5df28-3892-4295-a0eb-fbc668882171. Error creating salesOrder. CS error: Unable to generate the next number. Manual numbering is activated for 'SOORDER'." }
[/CODE]
```