PrintVis

Partner Training 2023

Table Of Contents

Who are we?

Agenda

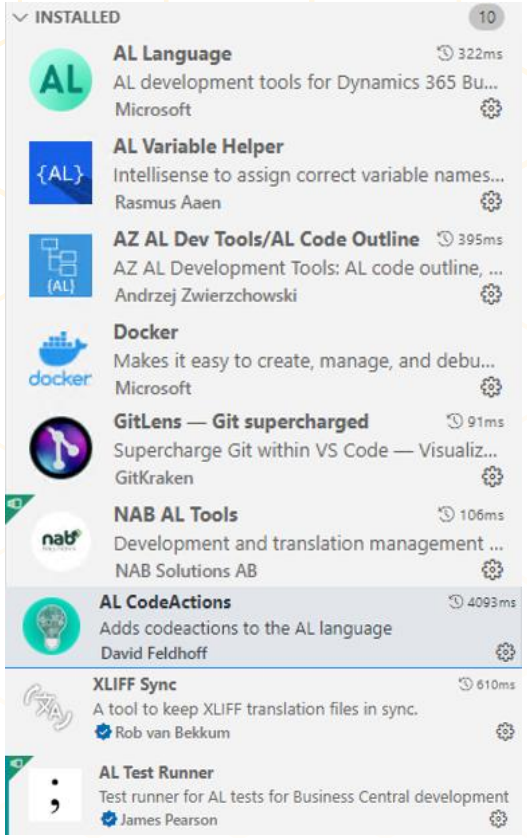
Get started

Hands on

Plan

	8:30am-10:00am	10:15am-11:45am	12:00pm-1:00pm	1:00pm-2:30pm	2:45pm-5:30pm
Tuesday	Intro, Demo, Status Code, Calculation formulas, Unit conversion, Userfields	Invoice template	Lunch	ShopFloor Metro, Job Ticket	Upgrades
Wednesday	Create case with code	Cloud migration	Lunch	Open session for specific questions	
Thursday	One-on-one meetings with a PrintVis developer		Lunch	One-on-one meetings with a PrintVis developer	

VS code Extensions



Good looking names on variables

New objects

Makes working with Git easy

Translation tool

Run tests in VS Code

Event Requests

- We are extending PrintVis all the time with new events
- A new version is shipped every week
- Use the word document to request
- We are approving or get back with an alternative solution asap.

- If other changes are needed, create a support ticket and explain why.
  (we will fix it or guide you to a solution)
  - Local procedures => non-local etc..
  - Adding procedures to PrintVis objects
  - Access to global variables

Bug/issue Reporting

- If you find something that could be a potential bug report it to support365.printvis.com
- Provide as much data as possible, debugger screenshots etc.. Solution proposals.
- If critical, new release ASAP
- If not crititical next Tuesday/wendesday
- Support is monitored all business hours (europe) by developer

Demo Data

Demo Data

Trial Data

Trial data can be imported from the *Configuration Packages* page by clicking on the *PrintVis Import and Apply Trial Package* action. It is a smaller demo set that can be used as a sample setup for prospective clients. This does not display all functionality but includes standard commercial print cost centers and sample cases.

Demo database PrintVis 21 (Business Central 21 based)

Download the release database which contains 7 demo-companies:

- PrintVis Unlimited
- PrintVis Inc.
- PrintVis Bindery
- PrintVis Label
- PrintVis Newspaper
- PrintVis Packaging
- PrintVis JDF Unlimited

Getting started

- Docker containers with traefik
  - Launch files
  - Script used
  - VS Code Extensions ->
- Create extension depending on PrintVis
  - Add something and publish
  - Add dependencies to PrintVis

- Use prefix on all, objects, fields,names
- If ID is used, put in 50000..99999 (even if its not mandatory)

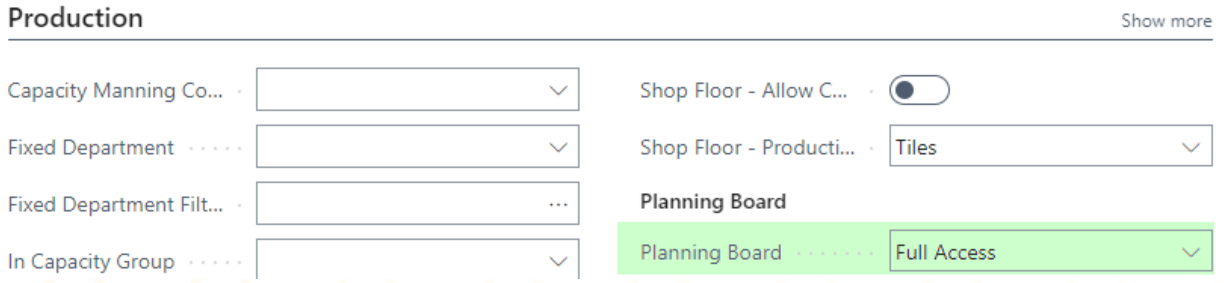
Demo flow – need to know level

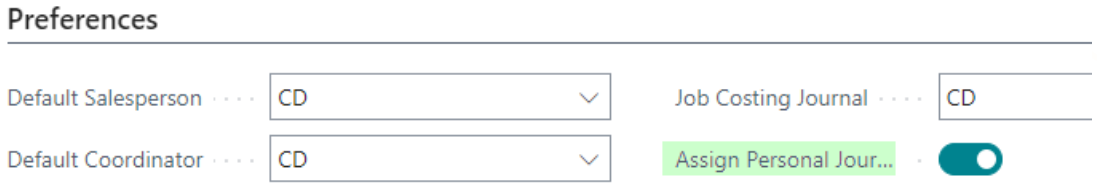
- Initial setup
- Estimate
- Quote
- Order
- Material requirement
- Planning
- Planning board
- Shop floor
- Time registration
- Material registration
- Shipment
- Invoicing

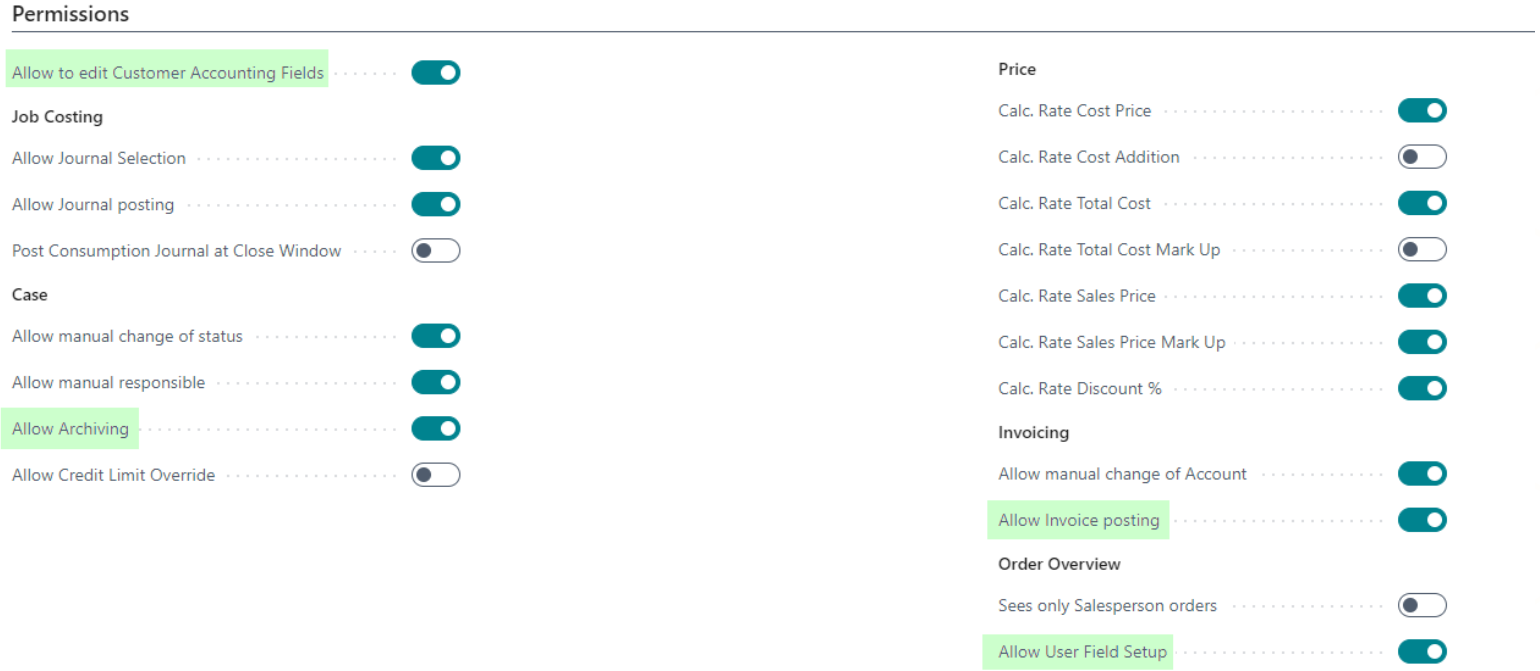
Goal:
To be able to follow the flow

PrintVis User

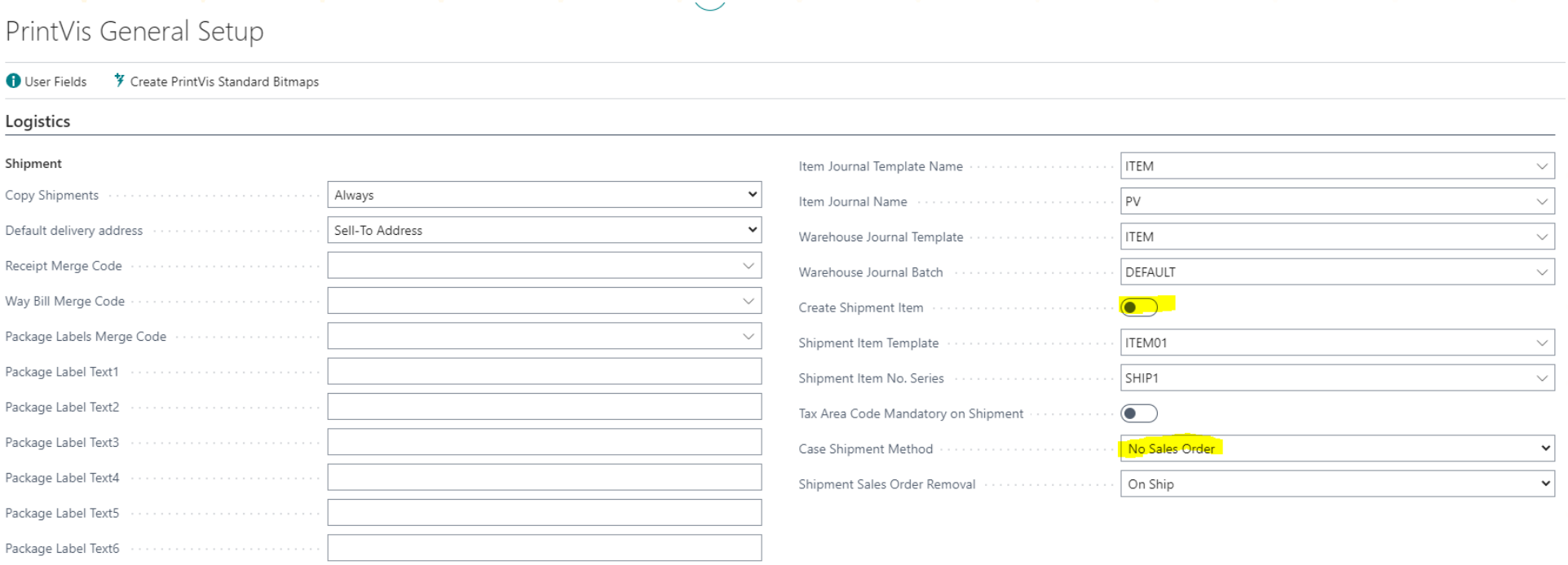
- Change profile to PrintVis Admin
- Create a new PrintVis User
  - Enable: Coordinator, Estimator, Salesperson, Capacity Resource, Assign personal Journal, Allow Archiving, Allow User Field Setup
  - Planning Board: Full Access





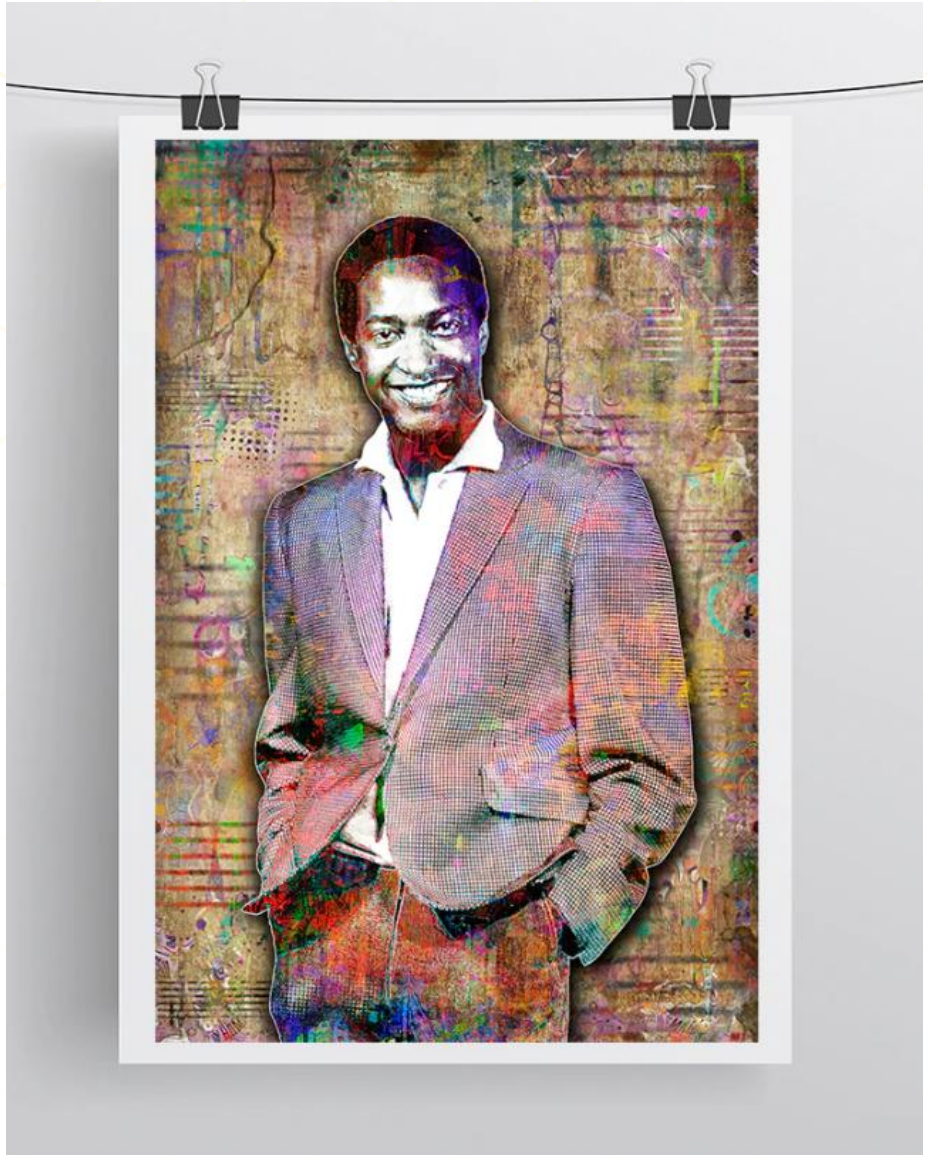


PrintVis General Setup



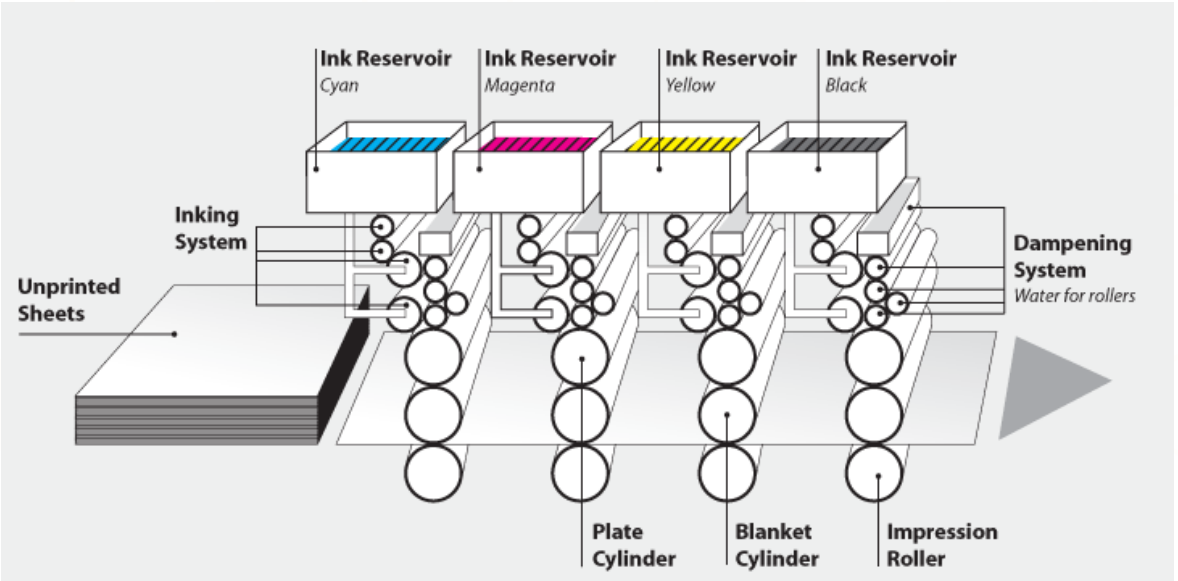
The Product

- Poster
- Size
  - 50 inches width
  - 70 inches height
- Material paper
  - PS-436
  - Finishing 7820
- 4+0 colors



The Process

- Prepress 2 h
- 4 plates
- Ink
- Paper
- Trimming



PrintVis Case

- Case management
- New Case
  - Select Customer
  - Order type DIRECT
  - Product Group POSTER (130)

This is like the sales order
Describes the business

Format Code 70x50
Width x Length PrintVis General Setup
What unit? See PrintVis Standard Units

- Job line
  - Quantity 10000
  - Pages 1
  - Format Code 70x50
  - Colors front 4

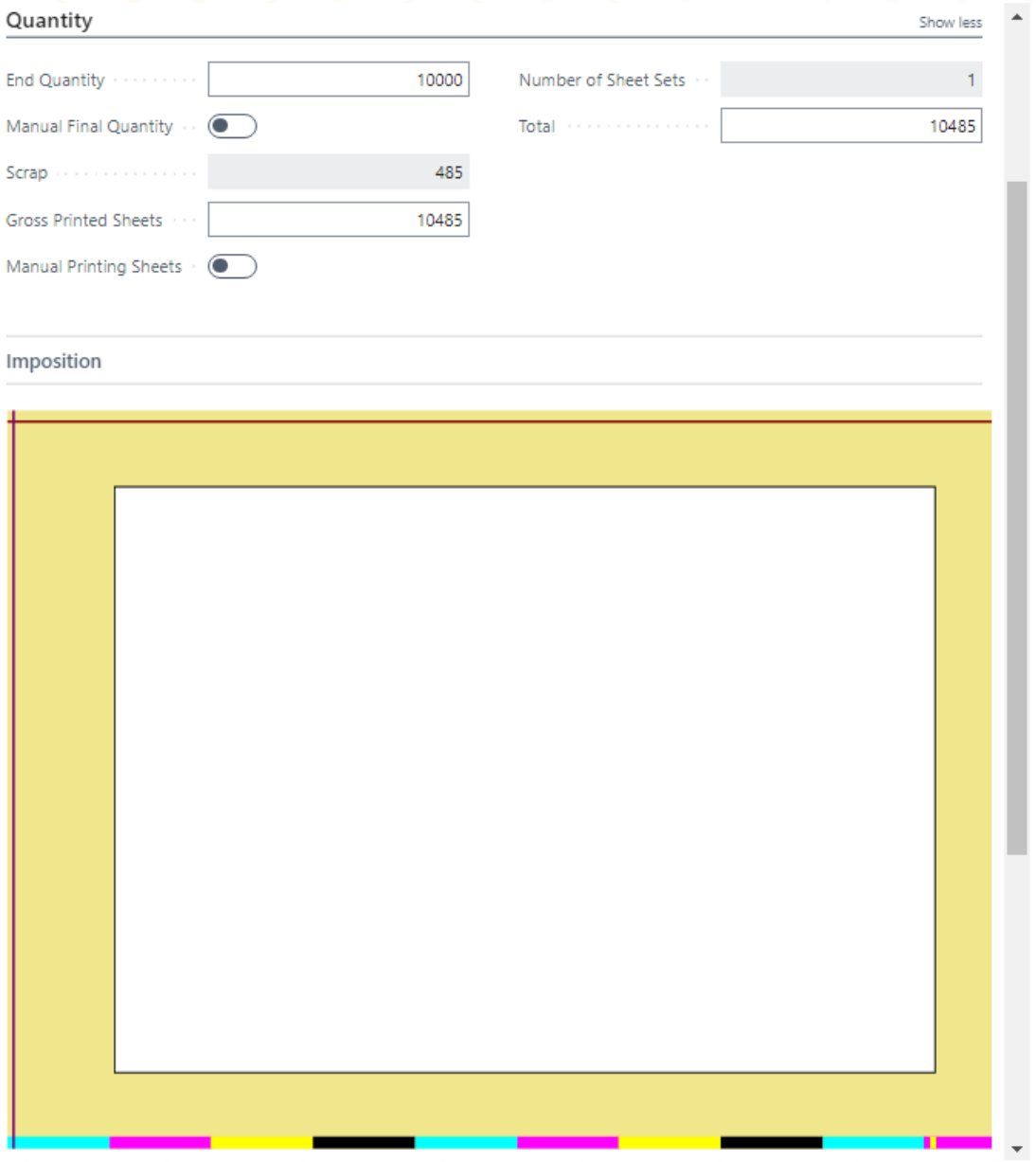
PrintVis Standard Units

Search | New | Edit List | Delete | Metric (CM) | Metric (MM) | US Standard

Type	Description	Formula Text	Format Factor	Per Paper Weight
Formula	cm		1	
Margins	mm	cm * 10	10	
Reel Length	m	cm / 100	0.01	
Paper Weight	Grammage	Square cm / 1,000 / (100*100)	0	
Paper Thickness	Bulk	cm / Paper Weight * 10,000	10,000	☑
Paper Area	sqm	Square cm / 10,000	0.0001	

PrintVis Job Card

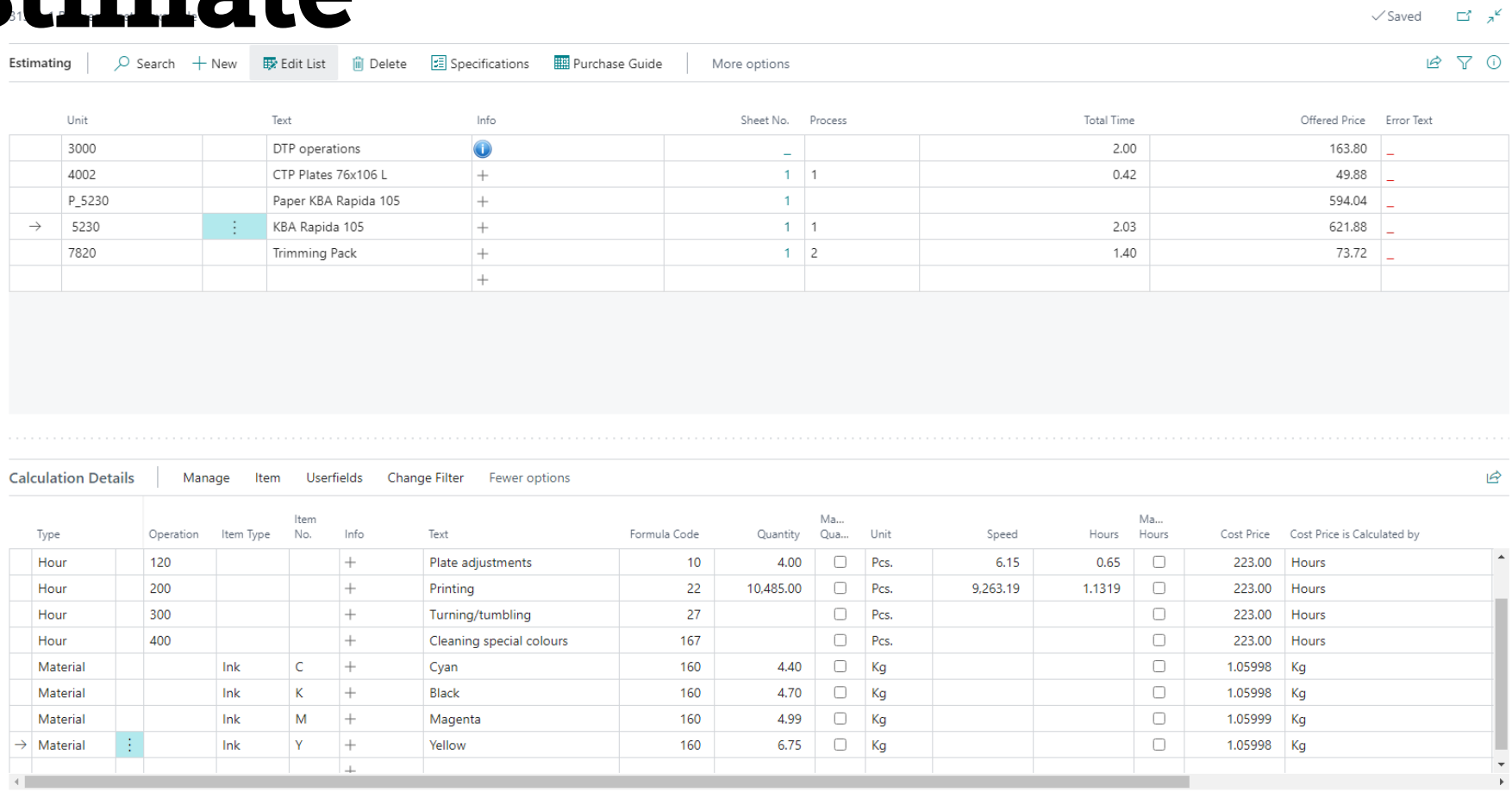
- Job line
  - Click Job Card
    - Select Paper PS-436
    - Select Machine L_5230
    - Add finishing - Optional
    - Check specification



PrintVis Estimate

- DTP operation
  - Add hours on a line
- Sheet Offset Printer
  - Notice all processes

Estimate contains all time and material for the job



PrintVis Quote

- Case card
  - Change status to Quote
  - Report -> Print/Send -> Quote

Job line is changed from Request to Quote

Quote

612 South Sunset Drive
Spotsmeyer's Furnishings
Miami, US-FL 37125
USA

Quote No.	Quote Date	Your Reference	Responsible
Q-000098	February 18, 2023		Thomas Beck Andersen

We thank you for your inquiry and hereby take pleasure in offering you our best Price and Services for the following job specification:
Job Name: Poster example

Quantity	Description	Unit Price	Total price in USD
10000	10 000 Posters, size 50X70 in 4+0 colours Pages: Text 1 Size: Text 50X70 Paper: Text Tauro Offset Weight:90.00 Offset: L_5230 Colors: Text 4/0 Finishing: Trimming	0.15	1,503.16

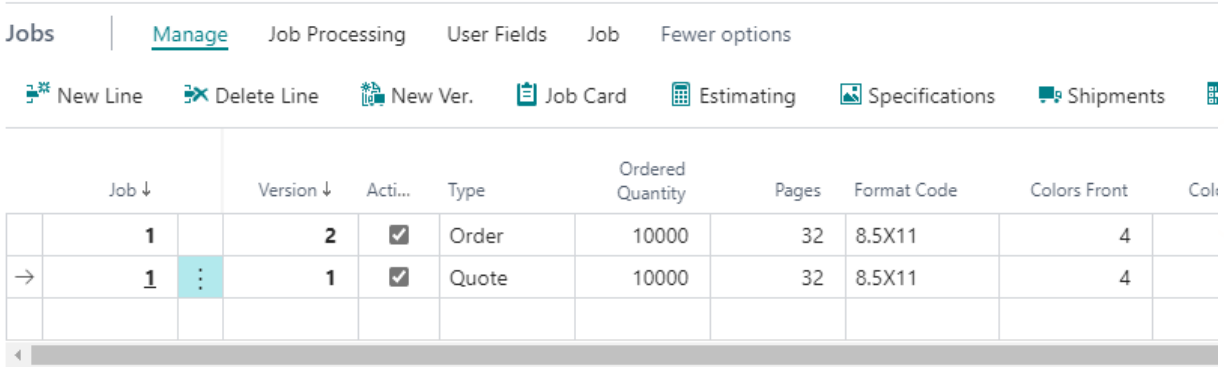
We sincerely hope that our quotation will meet your expectation and look forward to serving you in this matter.

Best Regards,
John Roberts

PrintVis Order

- Case card
  - Set a requested Shipment date on job
  - Change status to PLAN

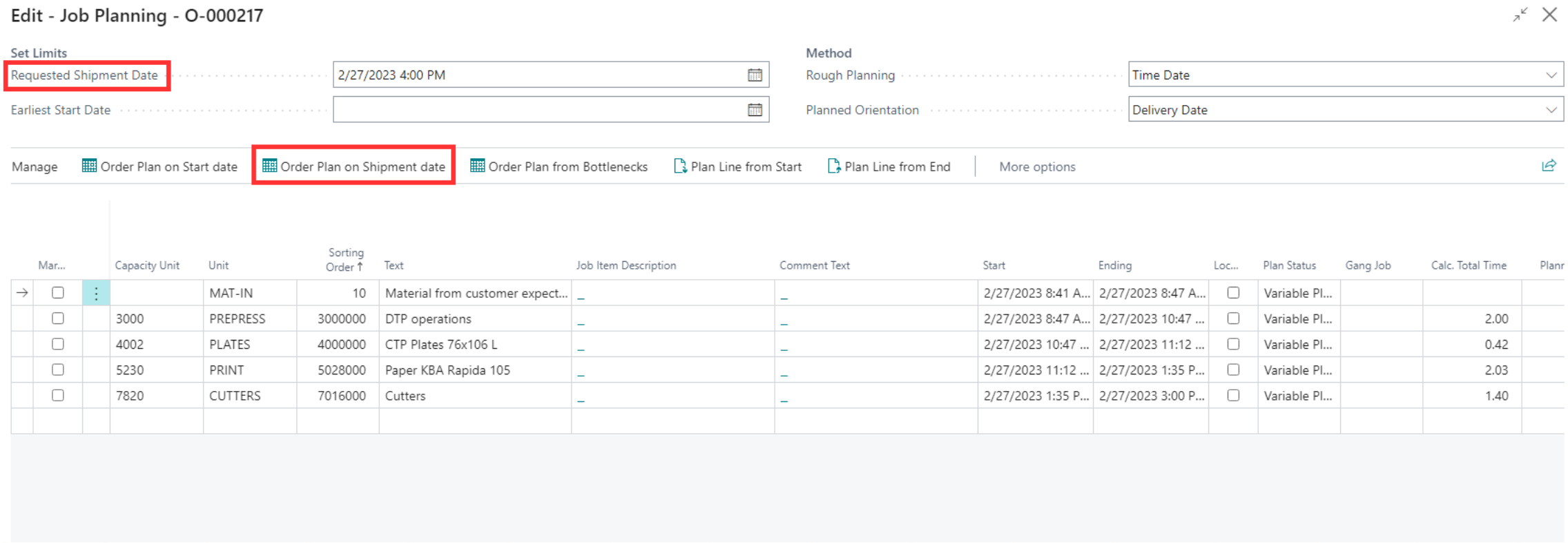
New job line created Order



Job	Version	Acti...	Type	Ordered Quantity	Pages	Format Code	Colors Front
1	2	☑	Order	10000	32	8.5X11	4
1	1	☑	Quote	10000	32	8.5X11	4

PrintVis Planning

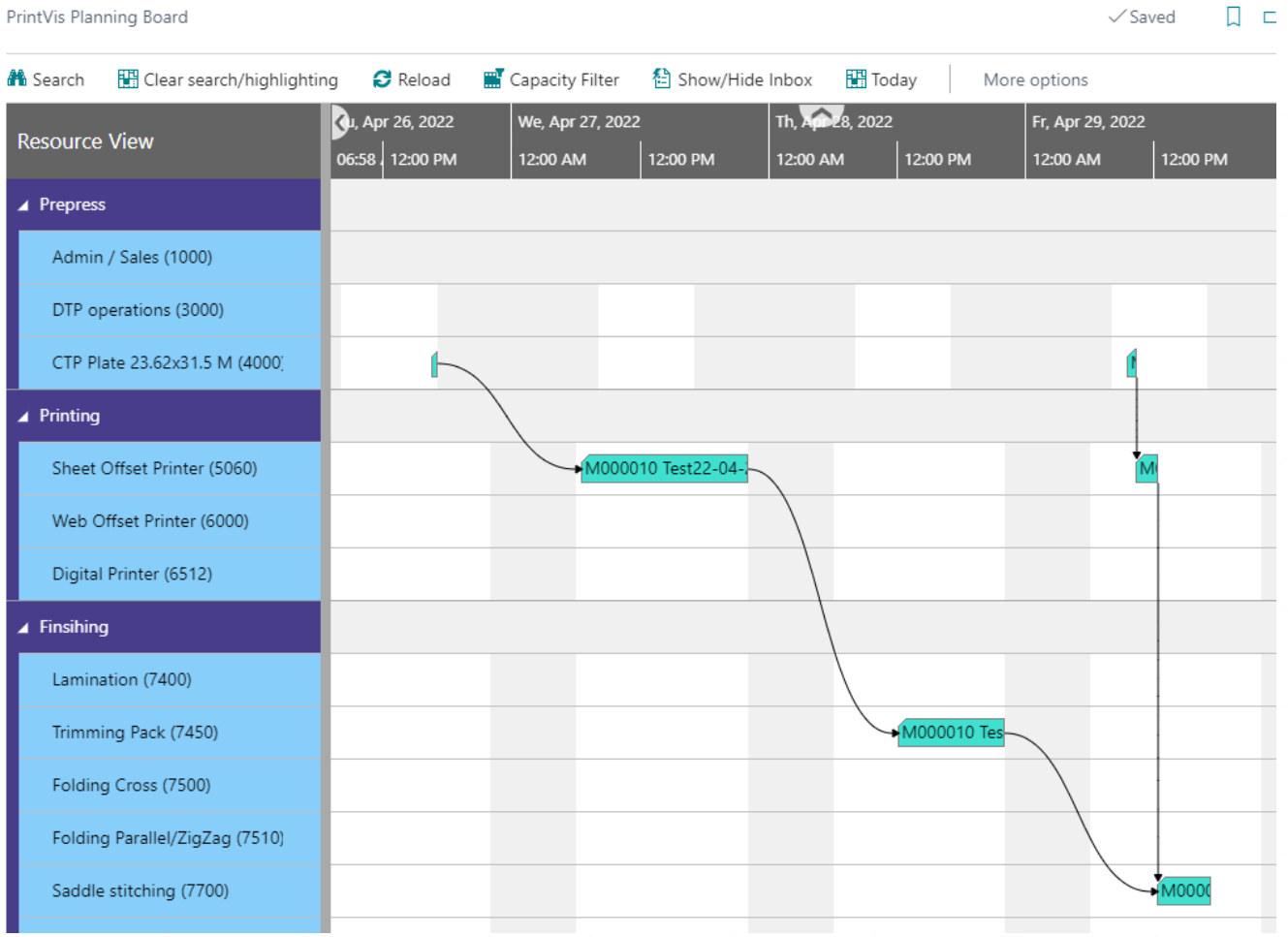
- Job Line
  - Planning
  - Plan from shipment date



PrintVis Planning Board

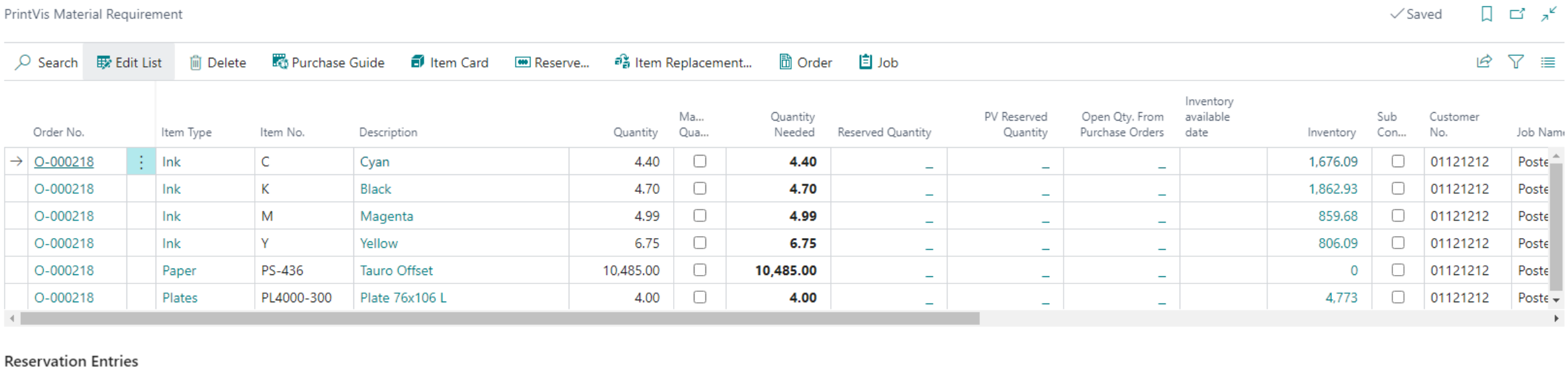
- Planning board

Search For PrintVis Planning Board



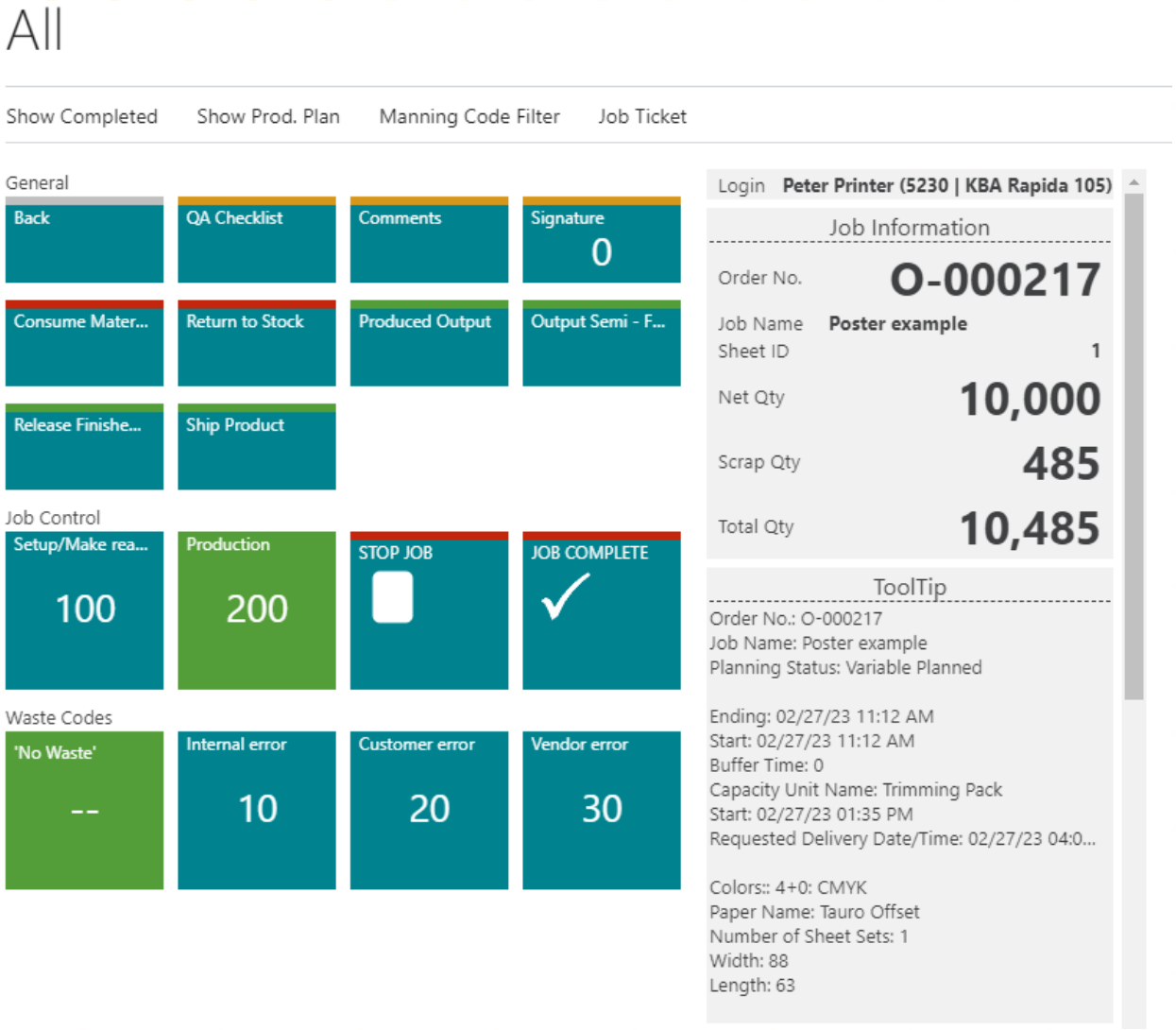
PrintVis Material Requirement

- Case Card
  - Material Requirement
  - Shows the requirement for the case



PrintVis Shop Floor

- Change Role to PrintVis Shop Floor Worker
  - Login with PPR
  - Select Cost Center Sheet Offset Printer
  - Find the job and start production



PrintVis Shop Floor

- Consume material
  - Consume some paper
- Click stop
  - Time is registered
  - Complete the job
- Log out!

PrintVis Shipment

- Change back Role to PrintVis Admin or PrintVis Coordinator
  - Open the case card again
  - Click on Shipments
  - Put in quantity to ship
  - Ship
  - Change status to INVOICE

- Methods
  - Create sales order per shipment
  - Just mark it as shipped

PrintVis Invoice

- Form the case card
  - Click on Invoice Guide
  - Click new
  - Click build
  - Post

Workshop run the demo flow

- Run through the demo flow
  - When you have a posted invoice, you are done.

Data Structure

The PrintVis Base App has around 250 tables which are a mixture of setup and transactional data.

PVS Case

PK: ID

Table:PVS Case	
Key field:ID	
Commonly used Fields	
Sell To No.	Who is ordering
Job Name	Text field for describing the case
Status Code	Current Status of the Case
Product Group and Order Type	Define type of case

The most central table in the system. Think of the case as a placeholder for all information about a particular customer task.
The primary key for the case is simply an entry no. called “ID”. During its lifecycle the case will change status, starting off as “request.” When the case changes status to “Quote” it will get a quote No. assigned (from a no. series) and when the case change status to “Order” it will get a Order No. assigned (from a no. series).

When there are no activities expected on a case it will be marked as “Archived” (a simple Boolean field) and assigned an “Archived Date.” This is typically done when an order is invoiced, or a quote is rejected. This can be reverted, and the case can again be removed from archive (simply by setting the Boolean field “Archived” to false)

PVS Job

PK: ID, Job, Version

The job table is the main placeholder for describing the product that we want to produce.
In many companies you only have one job per case, whereas other companies are used to working with several jobs per case. This is a matter of company policies.
The primary keys for the job are ID, Job, Version. Where the ID is the reference to the case, Job is the job no. and version gives you the opportunity to have several versions of the job.
Only one if these versions will be the active version that is going to be produced.

ID	Job	Version	Status	Active	Quantity
123	1	1	Quote	Yes	5000
123	1	2	Quote	No	10000
123	1	3	Quote	No	15000

```
PVSJob.Setrange(ID,123);
PVSJob.Setrange(Active,true);
PVSJob.Setrange("Quote Calculation",true);
if PVSJob.FindSet() then...
```

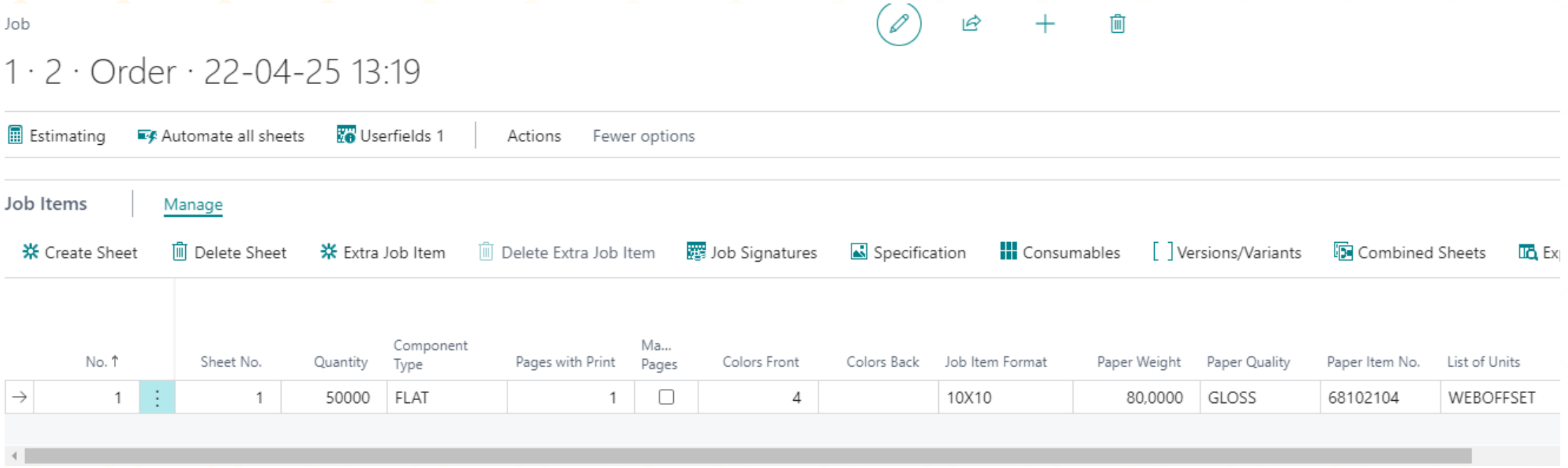
ID	Job	Version	Status	Active	Quantity
123	1	1	Quote	No	5000
123	1	2	Quote	Yes	10000
123	1	3	Quote	No	15000
123	1	4	Order	Yes	7000

Table:PVS Job	
Key field:ID, Job, Version	
Commonly used Fields	
Quantity	Ordered/estimated quantity
Quoted Price	Quoted price for the line
Active	Check if the line is active
Requested Delivery Date	Requested Delivery Date

PVS Job Item

PK: ID, Job, Version, Job Item No., Entry No.

The job item tables is used to describe the different parts of a product.
Book = Cover + Text part.
It's here where the main machine is connected to the job.

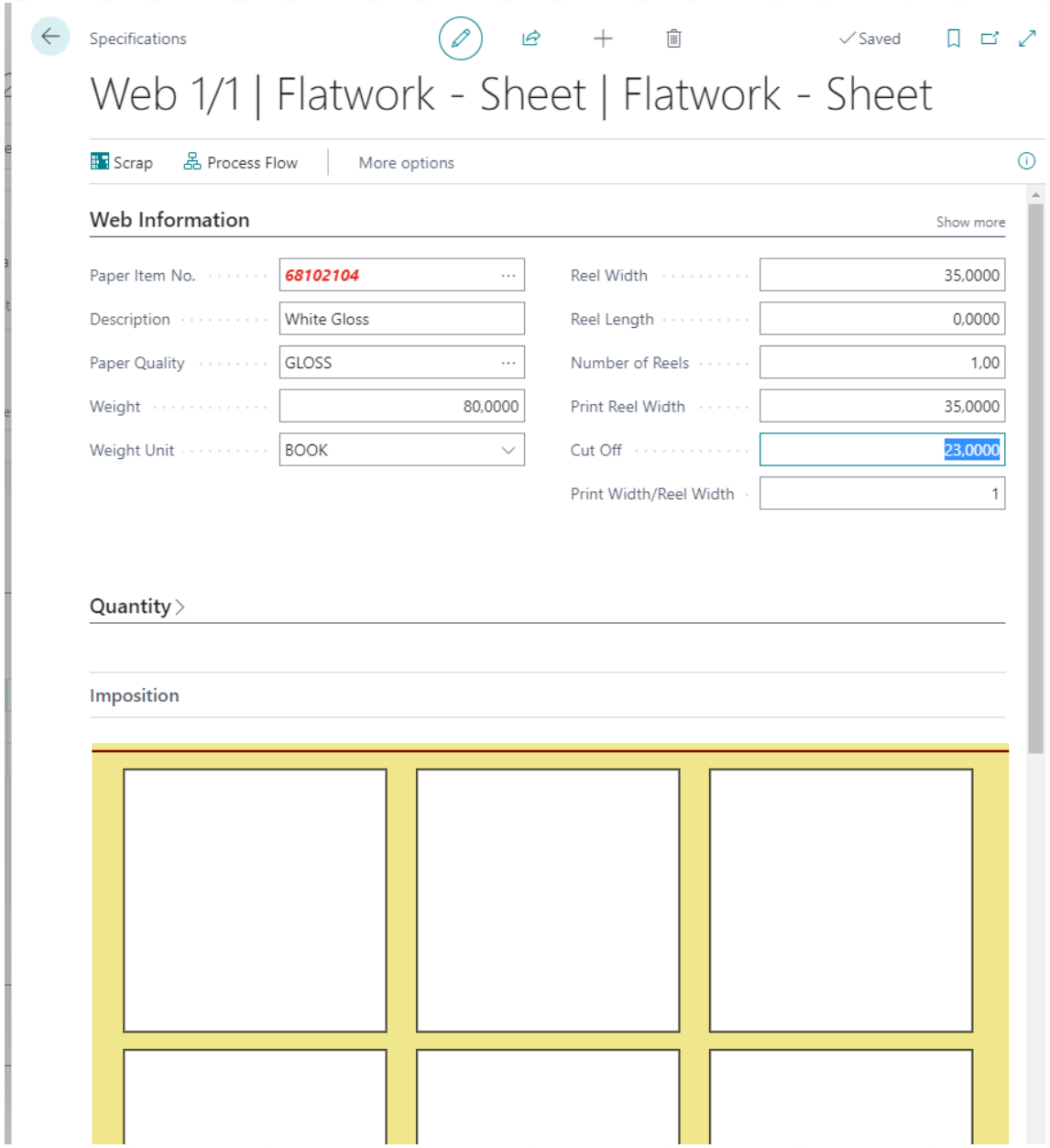


PVS Sheet

PK: Sheet ID

The Sheet table stores information about the sheet, on the sheet the Job Item is produced/printed.

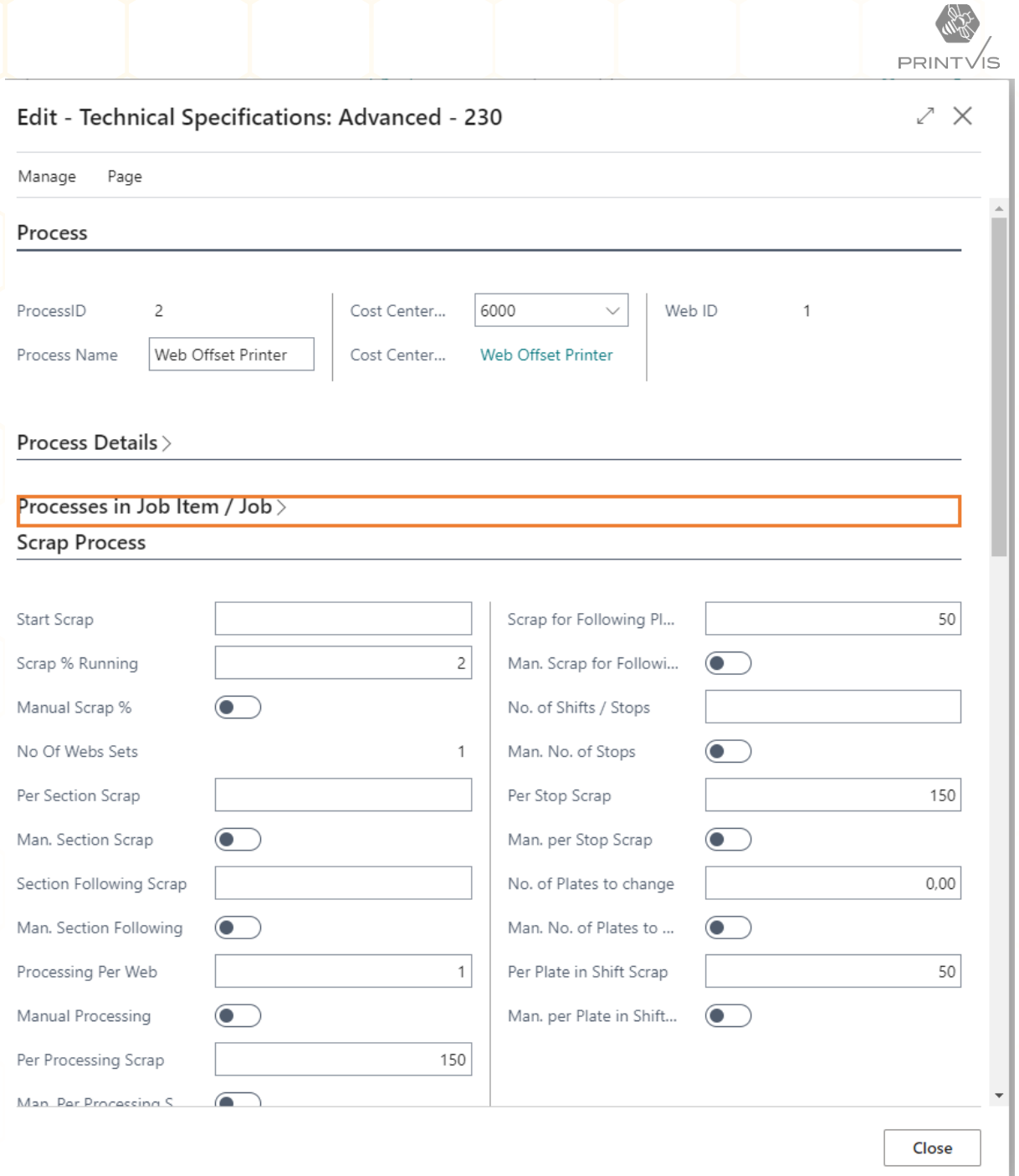
Information that is useful from here could be
Size of the sheet. Quantity needed for producing the product.



PVS Job Process

PK: Process ID

The Job process tables stores information about the different processes, scrap is probably the most important thing to mention here.

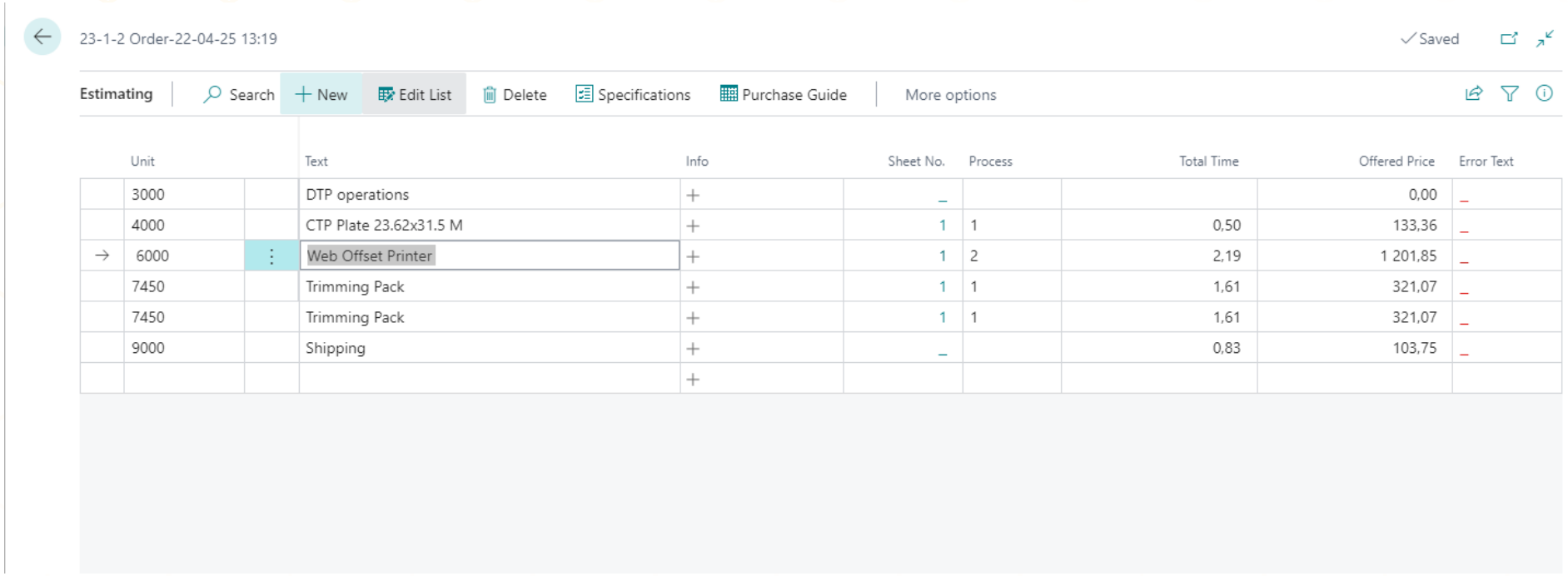


PVS Job Calculation Unit

PK: ID, Job, Version, Entry No.

A puzzle piece of the estimation. The Calculation unit could be connected to a Sheet and a Process.

The Calculation Unit is also connected to a Planning unit, with the field Plan ID



Unit	Text	Info	Sheet No.	Process	Total Time	Offered Price	Error Text
3000	DTP operations	+	_			0,00	_
4000	CTP Plate 23.62x31.5 M	+	1	1	0,50	133,36	_
→ 6000	Web Offset Printer	+	1	2	2,19	1 201,85	_
7450	Trimming Pack	+	1	1	1,61	321,07	_
7450	Trimming Pack	+	1	1	1,61	321,07	_
9000	Shipping	+	_		0,83	103,75	_
		+					

PVS Job Calculation Detail

PK: ID, Job, Version, Entry No., Line Type, Unit Entry No.

The smallest puzzle piece of the estimate.
It could either be a material line or an hour line.

Material line:
Item
Quantity
Cost and sales prices

Hour line:
Operation (what to do)
Hours
Cost and sales prices

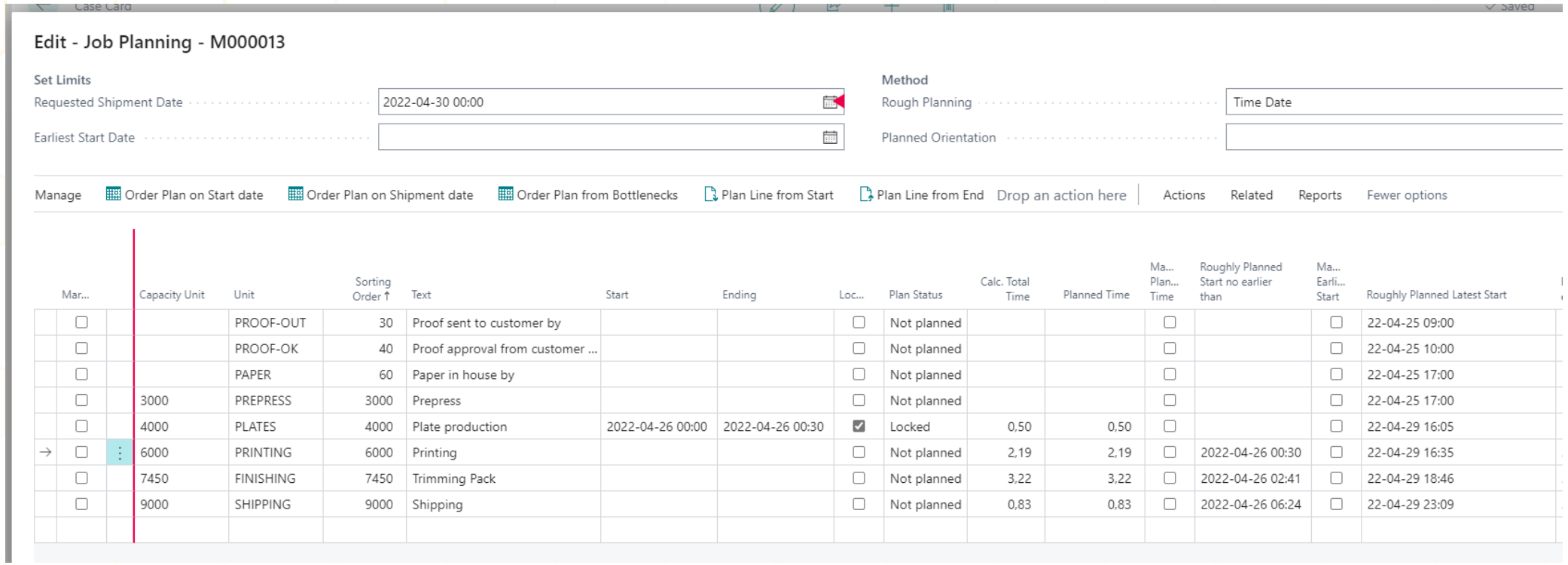
Both lines have also:
Cost Center – what machine/resource
Unit of measure – how to consume
Line Type – Normal or additional

PVS Job Planning Unit

PK: Simulation Version, ID, Job, Plan ID, Entry No.

Things that needs to be planned or are already planned. Planning against capacity units.

To get the Calculation details connected to the planning unit, the Plan ID is the field that exists in both tables.



Mar...	Capacity Unit	Unit	Sorting Order	Text	Start	Ending	Loc...	Plan Status	Calc. Total Time	Planned Time	Ma... Plan... Time	Roughly Planned Start no earlier than	Ma... Earli... Start	Roughly Planned Latest Start
☐		PROOF-OUT	30	Proof sent to customer by			☐	Not planned			☐		☐	22-04-25 09:00
☐		PROOF-OK	40	Proof approval from customer ...			☐	Not planned			☐		☐	22-04-25 10:00
☐		PAPER	60	Paper in house by			☐	Not planned			☐		☐	22-04-25 17:00
☐	3000	PREPRESS	3000	Prepress			☐	Not planned			☐		☐	22-04-25 17:00
☐	4000	PLATES	4000	Plate production	2022-04-26 00:00	2022-04-26 00:30	☑	Locked	0,50	0,50	☐		☐	22-04-29 16:05
→ ☐	6000	PRINTING	6000	Printing			☐	Not planned	2,19	2,19	☐	2022-04-26 00:30	☐	22-04-29 16:35
☐	7450	FINISHING	7450	Trimming Pack			☐	Not planned	3,22	3,22	☐	2022-04-26 02:41	☐	22-04-29 18:46
☐	9000	SHIPPING	9000	Shipping			☐	Not planned	0,83	0,83	☐	2022-04-26 06:24	☐	22-04-29 23:09

PVS Job Costing Journal/Line

PK: Simulation Version, ID, Job, Plan ID, Entry No.

When users reporting time or material everything ends up in this PrintVis Job Costing Journal. Then when the Journal is posted different thing will happen with lines of type time and material.

PVS Job Costing Labor Entry
Contains all posted time

PVS Job Costing Entry
Contains all posted time and material, has connection to Item Ledger Entry or PVS Job Costing Labot Entry. Also contains information of posted invoices.

PVS Shop Floor Journal Entry

PK: Simulation Version, ID, Job, Plan ID, Entry No.

Used on shopfloor to track ongoing time reporting.

Also used to collect the reporting from shop floor if realtime posting is deactivated.

PVS Template

Template is just a normal case.
On the Job line its possible to mark it as a template.
Template could be connected to Product groups or Items.

Workshop Status Code

- Status Codes are used on Cases
- Triggers functionality / checking data
- Used in all installations

- What to extend?
- Check data before status change
- Modify data after status change

1. Before status change to Quote, check Quoted Price on the Job, if > 1000 then error out
2. Create a table + List Page
    1. PTE Status Change Log
        1. ID PK
        2. Entry No. PK
        3. Old status Code[50]
        4. New status Code[50]
        5. On the page add system field modified at
    2. Add record after status code is changed

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Main", 'OnCaseBeforeStatusChange', ''

[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Main", 'OnCaseAfterStatusChange', ''
```

Workshop User Field 1/2

- Can be set for different tables
  - PVS Case, PVS Job, Customer, etc.
- Can be grouped and assigned to a PrintVis Department
- Can be used in
  - Calculation formulas
  - Merge fields
  - Printed reports

Use the **PVS Userfield Management** codeunit

```
UserfieldManagement.Get_UserField_Value(…);

UserfieldManagement.Set_UserField_Value(…);
```

Workshop User Field 2/2

- Task 1
  - Create user field on case level
  - Extend Report PVS Word Dataset Case Quote (6010405) with the created user field
    - Create custom layout and add the field
    - Run the report with the custom layout
- Task 2
  - Create user field on job level
  - Modify the user field with the current date time when Case status changes to Plan
    - Subscribe to `OnCaseBeforeStatusChange` event in PVS Main codeunit
  - Add the userfield value to the job list part on the case card.

Workshop Calculation formula 1/3

- Could be used in several places
  - Estimate
  - Machine setup
  - Speed / scrap tables
  - Pricelists
- Triggers functionality / checking data
- Used in all installations

```
codeunit 50151 "PTE Calculation Formula"
{
    trigger OnRun()
    begin
        SingleInstance.Get_Current_CalcUnitDetailRec(JobCalculationDetail);
        JobCalculationDetail."Qty. Calculated" := 0; // Clear the field to be calculated
        JobCalculationDetail."Qty. Calculated" := Random(10000); // Assign value
        SingleInstance.Set_Current_CalcUnitDetailRec(JobCalculationDetail); // Push back the result
    end;

    var
        JobCalculationDetail: Record "PVS Job Calculation Detail";
        SingleInstance: Codeunit "PVS SingleInstance";

}
```

Workshop Calculation formula 2/3

Calculation Formulas

- Some data is calculated on temp tables
  - Calculation Details

    ```
    SingleInstance.Get_CalcUnitDetailsRecTmp(JobCalculationDetailTemp); //read calculation lines
    ```
  - Sheet

    ```
    SingleInstance.Get_SheetRecTmp(JobSheetTemp); // Use this if you need info from the sheet
    ```
  - Process

    ```
    SingleInstance.Get_ProcessRecTmp(JobProcessTemp); // Need info from the process
    ```
- The rest is possible to read as normal

```
codeunit 50152 "PTE Calculation Formula 2"
{
    trigger OnRun()
    begin
        SingleInstance.Get_Current_CalcUnitDetailRec(JobCalculationDetail);
        SingleInstance.Get_SheetRecTmp(JobSheetTemp);
        JobCalculationDetail."Qty. Calculated" := 0; // Clear the field to be calculated
        JobSheetTemp.SetRange("Sheet ID", JobCalculationDetail."Sheet ID");
        if JobSheetTemp.FindFirst() then
            JobCalculationDetail."Qty. Calculated" := JobSheetTemp.Weight;
        SingleInstance.Set_Current_CalcUnitDetailRec(JobCalculationDetail); // Push back the result
    end;

    var
        JobSheetTemp: Record "PVS Job Sheet" temporary;
        JobCalculationDetail: Record "PVS Job Calculation Detail";
        SingleInstance: Codeunit "PVS SingleInstance";
```

Workshop Calculation formula 3/3

Calculation Formulas

- Task 1
  - Create formula 1010 that returns the weekday (DATE2DWY(TODAY(), 1);)
- Task 2
  - Create formula 1011 that returns the area of the printed sheet
    - Sheet.width x Sheet.length
- Create both formulas in same codeunit

```
case JobCalculationDetail."Formula Code" of
    1010:Formula_1010_Weekday();
    1011:Formula_1011_Area();
end;
```

Workshop Shop Floor 1/3

Shop Floor

- Most common is to add an extra tile
- Two events are used
  - One for adding the tile

    ```
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Shop Floor JS Add-in Mgt", 'PVSAddMoreTilesJobCard'
    ```

  - One for processing the click

    ```
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Shop Floor JS Add-in Mgt", 'OnAfterClickTileJobTicket'
    ```

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Shop Floor JS Add-in Mgt", 'PVSAddMoreTilesJobCard', '', false, false)]
local procedure PVSAddMoreTilesJobCard(var TempPVSSortingBuffertTiles: Record "PVS Sorting Buffer"; var isHandled: Boolean);
begin
    TempPVSSortingBuffertTiles.Init();
    TempPVSSortingBuffertTiles.PK1_Integer1 := 1;
    TempPVSSortingBuffertTiles.Text1 := '#CUSTOMSHIP#';
    TempPVSSortingBuffertTiles.Text2 := 'Shipments';
    TempPVSSortingBuffertTiles.Insert();
    isHandled := true;
end;
```

```
OnAfterClickTileJobTicket
Check inControlID is the same as the value in Text1 ‘'#CUSTOMSHIP#’
          Perform logic
In the end deselect the button with ReturnFlag := 2;
```

Workshop Shop Floor 2/3

Shop Floor

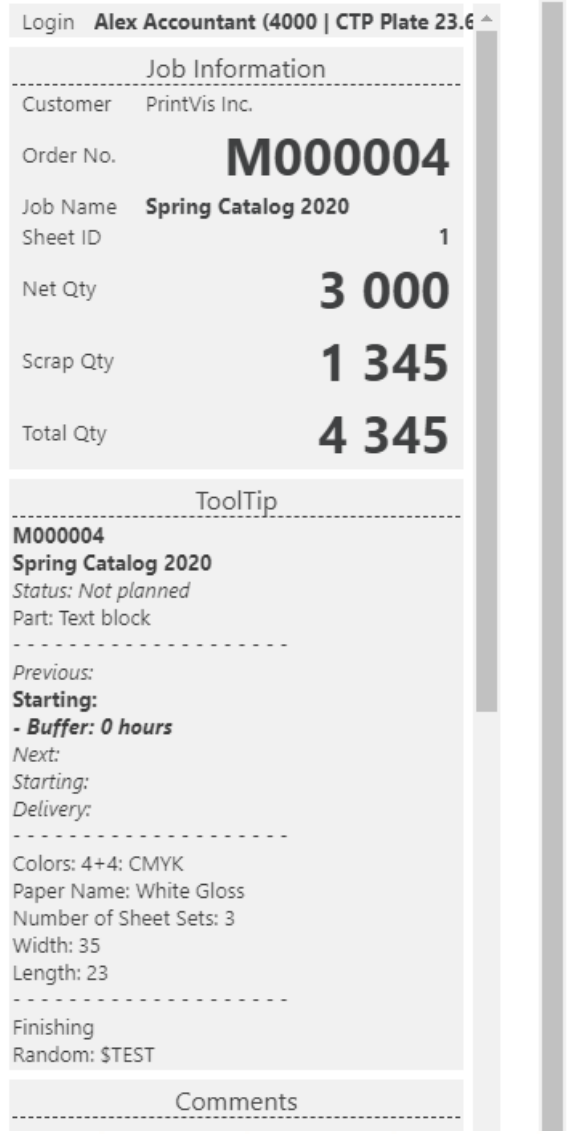
- Another change could be that the customer would like to have more info on right side

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Shop Floor JS Add-in Mgt",
'OnAfterBuildPanelBuffer', '', false, false)]
```

```
SF_FIELDINFO: label 'FIELDINFO', Locked = true;

    if pagename <> 'JobCard' then
        exit;

    PlanRec.SetPosition(JSAddinMgt.DecodeBase64ElementName(in_ControlID));
    PlanRec.Find('=');

    CaseRec.Get(PlanRec.ID);
```

```
in_PanelBuffer.Reset();
in_PanelBuffer.SetRange(Code1, SF_FIELDINFO);
if in_PanelBuffer.FindFirst() then begin
    TempSortingBuffer := in_PanelBuffer;
    TempSortingBuffer.Text1 := 'Customer';
    TempSortingBuffer.PK1_Integer2 := 1;
    TempSortingBuffer.Text2 := CaseRec."Sell-To Name";
    in_PanelBuffer.Init();
    in_PanelBuffer := TempSortingBuffer;
    if not in_PanelBuffer.Insert() then
        in_PanelBuffer.Modify();
end;
```

```
in_PanelBuffer.reset;
Page.Run(Page::"PTE Sorting Buffer", in_PanelBuffer);
```



Workshop Shop Floor 3/3

Shop Floor

- Task 1
  - Add a tile to open the shipment list filter on the Case ID
- Task 2
  - Add customer name in panel on right side
- Task 3
  - When shopfloor start and stop, send a Post to webhook.site with the Cost Center, Order No and Started/Stopped as text or json

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Shop Floor Management", 'OnBeforeJournal_Special_Start_Stop', '', false, false)]


        local procedure sendToSomewhere(TextToSend: Text)
        var
            myHTTPClient: HttpClient;
            myResponseMessage: HttpResponseMessage;
        begin
            myHTTPClient.Get('https://webhook.site/9113df2f-c59d-4823-9408-610cf80774e6?string=' + TextToSend,
    myResponseMessage);
        end;
```

Workshop Invoice Template 1/6

- Invoice templates are used to generate invoices
  - Possible to add “custom” lines when invoice is built up
  - Possible to manipulate the invoice line before insert
  - Possible to replace with custom build

Workshop Invoice Template 2/6

Invoice Template

- Possible to add “custom” lines when invoice is built up

```
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Invoice Template Mgmt", 'BuildInvoiceTemplateLine', '', false, false)]
    procedure BuildInvoiceTemplateLine(var SalesHeader: Record "Sales Header"; var InvoiceTemplateHeader: Record "PVS Invoice Template Header";
var InvoiceTemplateLine: Record "PVS Invoice Template Line"; var Job: Record "PVS Job");
    var
        InvoiceTemplateManagement: Codeunit "PVS Invoice Template Mgmt";
        InvoiceTemplateLineTemp: Record "PVS Invoice Template Line" temporary;
    begin
        IF InvoiceTemplateLine."No." <> 'TEST' then
            exit;
        InvoiceTemplateLineTemp.init;
        InvoiceTemplateLineTemp.Type := InvoiceTemplateLineTemp.Type::"G/L Account";
        InvoiceTemplateLineTemp.Description := 'This line is created by special function';
        InvoiceTemplateLineTemp."Manual Description" := true;
        InvoiceTemplateLineTemp.Quantity := 123;
        InvoiceTemplateLineTemp."Unit Price" := 100;
        InvoiceTemplateLineTemp.Insert();
        //Insert Invoice line
        InvoiceTemplateManagement.Insert_Salesline_New(SalesHeader, InvoiceTemplateLineTemp, InvoiceTemplateHeader, Job,'','','','','');
    end;
```

Workshop Invoice Template 3/6

Invoice Template

- Possible to manipulate the invoice line before insert

```
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Invoice Template Mgmt", 'Insert_Salesline_New_OnBeforeModify', '', false, false)]
    local procedure Insert_Salesline_New_OnBeforeModify(var SalesHeader: Record "Sales Header"; var SalesLine: Record "Sales Line"; var
InvoiceTemplateHeader: Record "PVS Invoice Template Header"; var InvoiceTemplateLine: Record "PVS Invoice Template Line"; var Job: Record "PVS
Job");
    begin
        if InvoiceTemplateHeader.Code <> 'STD' then
            exit;
        if InvoiceTemplateLine.Type <> InvoiceTemplateLine.Type::"Main Invoice" then
            exit;
        SalesLine.Description := SalesLine.Description + 'X';
    end;
```

Workshop Invoice Template 4/6

Invoice Template

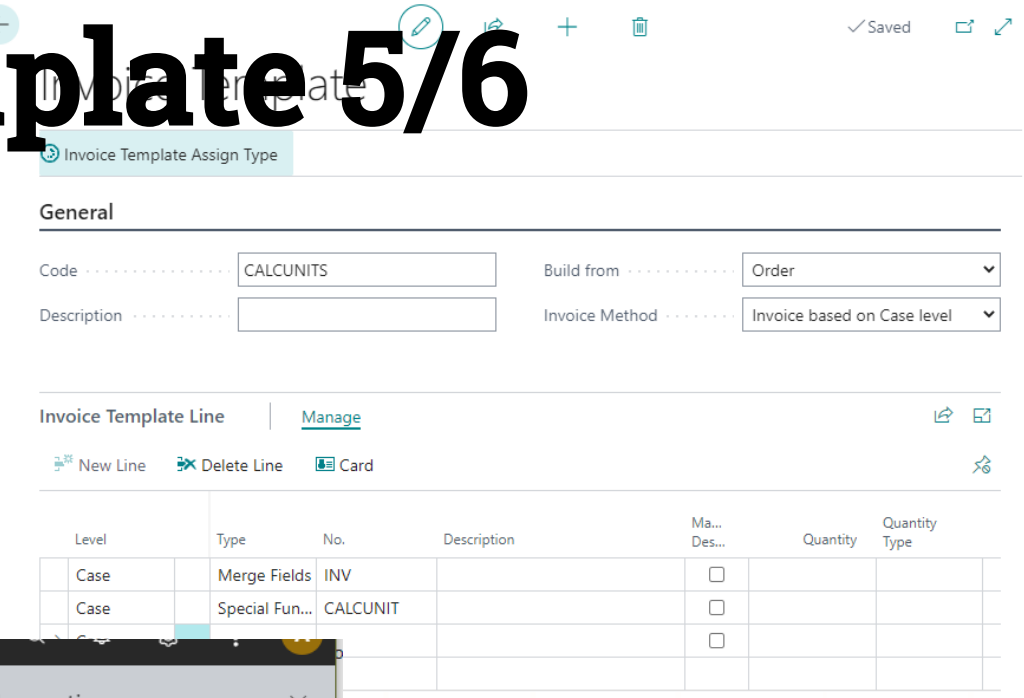
- Possible to replace with custom build
- The Case ID could be picked up from the Sales Header, apply it on all lines that are created

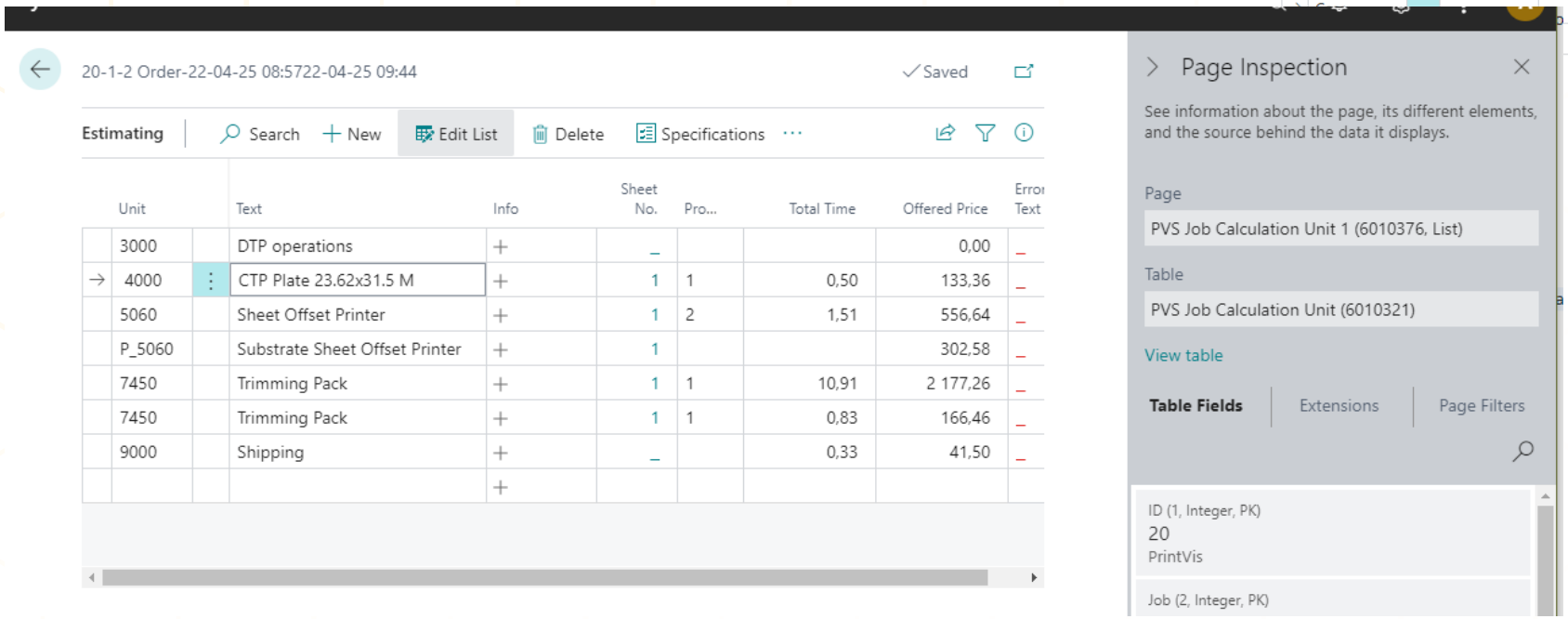
```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS Invoice Template Mgmt", 'BuildInvoiceTemplateHeader', '', false, false)]
local procedure BuildInvoiceTemplateHeader(var SalesHeader: Record "Sales Header"; var InvoiceTemplateHeader: Record "PVS Invoice Template Header");
var
    SalesLine: Record "Sales Line";
begin
    if InvoiceTemplateHeader.Code <> 'TEST' then
        exit;
    SalesLine.Init();
    SalesLine."Document Type" := SalesHeader."Document Type";
    SalesLine."Document No." := SalesHeader."No.";
    SalesLine."Line No." := 10000;
    SalesLine.Insert();
    SalesLine.Validate(Type, SalesLine.Type::"G/L Account");
    SalesLine.Validate("No.", '6110');
    SalesLine.Validate(Quantity, 123);
    SalesLine.Validate("Unit Price", 1);
    SalesLine."PVS ID" := SalesHeader."PVS Order ID";
    SalesLine.Modify(true);
end;
```

Workshop Invoice Template 5/6

Invoice Template

- Task 1
  - Build one invoice line per Calculation Unit
    Setup a new Invoice Template Line
    Find all calculation units for the Job
    Insert new Sales Line for each CalcUnit

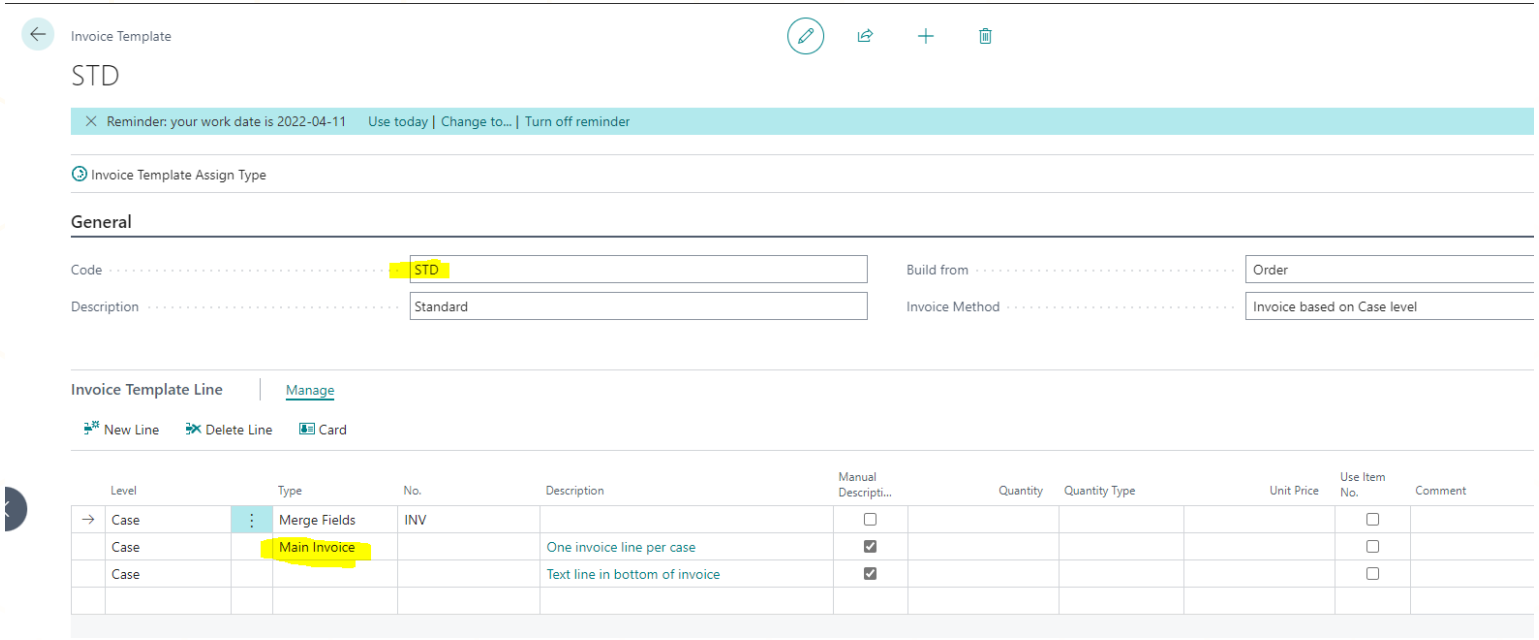




Workshop Invoice Template 6/6

Invoice Template

- Task 2
  - Round the unit price on the main invoice line.
    Find the template that has an invoice line of type “Main Invoice”
    Use Validate() to Round the Unit Price



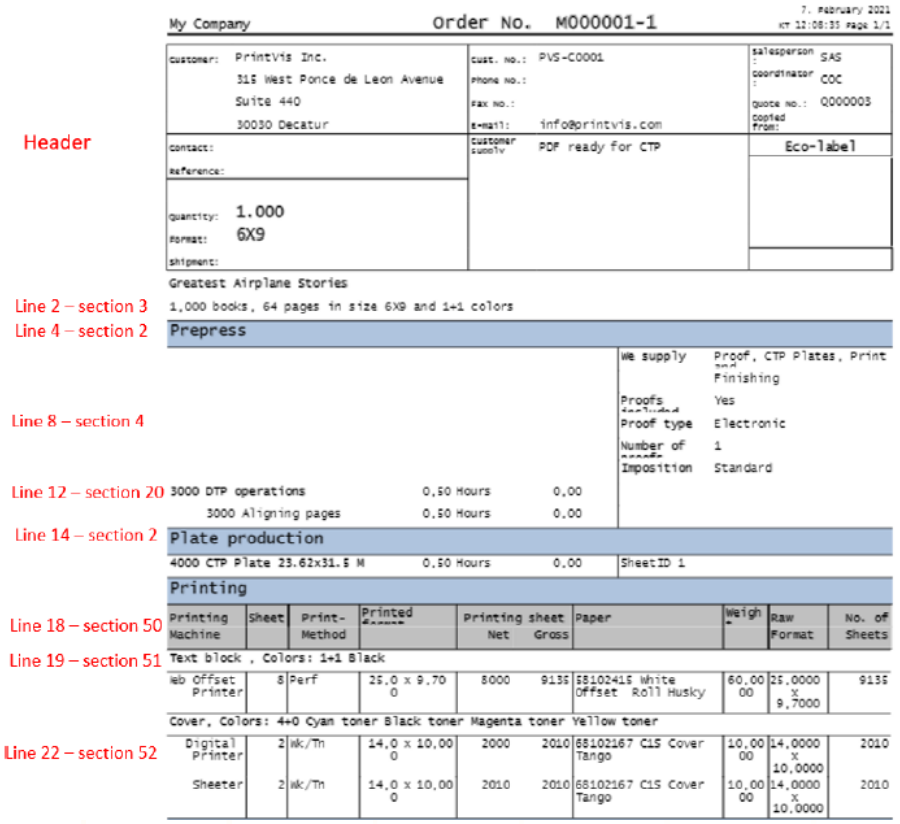
Workshop Job Ticket 1/8

- Most complex report in PrintVis
- Very often customized
- OLD - Dataset based on PVS Buffer Table
- New - temporary table PVS Job Ticket Temp

Set up which report to use on the PrintVis Report Selection Setup page

Workshop Job Ticket 2/8

OLD Job Ticket



My Company — Order No. M000001-1 — 7. February 2021 KT 12:08:35 Page 1/1

Header

customer: PrintVis Inc.
315 West Ponce de Leon Avenue
Suite 440
30030 Decatur

cust. no.: PVS-C0001
Phone no.:
Fax no.:
E-mail: info@printvis.com

salesperson: SAS
coordinator: COC
quote no.: Q000003
copied from:

contact:
reference:
customer supply: PDF ready for CTP
Eco-label

quantity: 1.000
format: 6X9
shipment:

Greatest Airplane Stories

Line 2 – section 3: 1,000 books, 64 pages in size 6X9 and 1+1 colors

Line 4 – section 2: **Prepress**

We supply	Proof, CTP Plates, Print and Finishing
Proofs included	Yes
Proof type	Electronic
Number of proofs	1
Imposition	Standard

Line 8 – section 4

Line 12 – section 20:

3000 DTP operations	0,50 Hours	0,00	
3000 Aligning pages	0,50 Hours	0,00	

Line 14 – section 2: **Plate production**

4000 CTP Plate 23.62x31.5 M	0,50 Hours	0,00	SheetID 1

Printing

Printing Machine	Sheet	Print-Method	Printed	Printing sheet Net	Printing sheet Gross	Paper	Weigh	Raw Format	No. of Sheets
Line 18 – section 50 / Line 19 – section 51: Text block , Colors: 1+1 Black									
Web Offset Printer	8	Perf	25,0 x 9,70 0	8000	9135	58102415 White Offset Roll Husky	60,00 00	25,0000 X 9,7000	9135
Cover, Colors: 4+0 Cyan toner Black toner Magenta toner Yellow toner									
Line 22 – section 52: Digital Printer	2	Wk/Th	14,0 x 10,00 0	2000	2010	68102167 CIS Cover Tango	10,00 00	14,0000 X 10,0000	2010
Sheeter	2	Wk/Th	14,0 x 10,00 0	2010	2010	68102167 CIS Cover Tango	10,00 00	14,0000 X 10,0000	2010

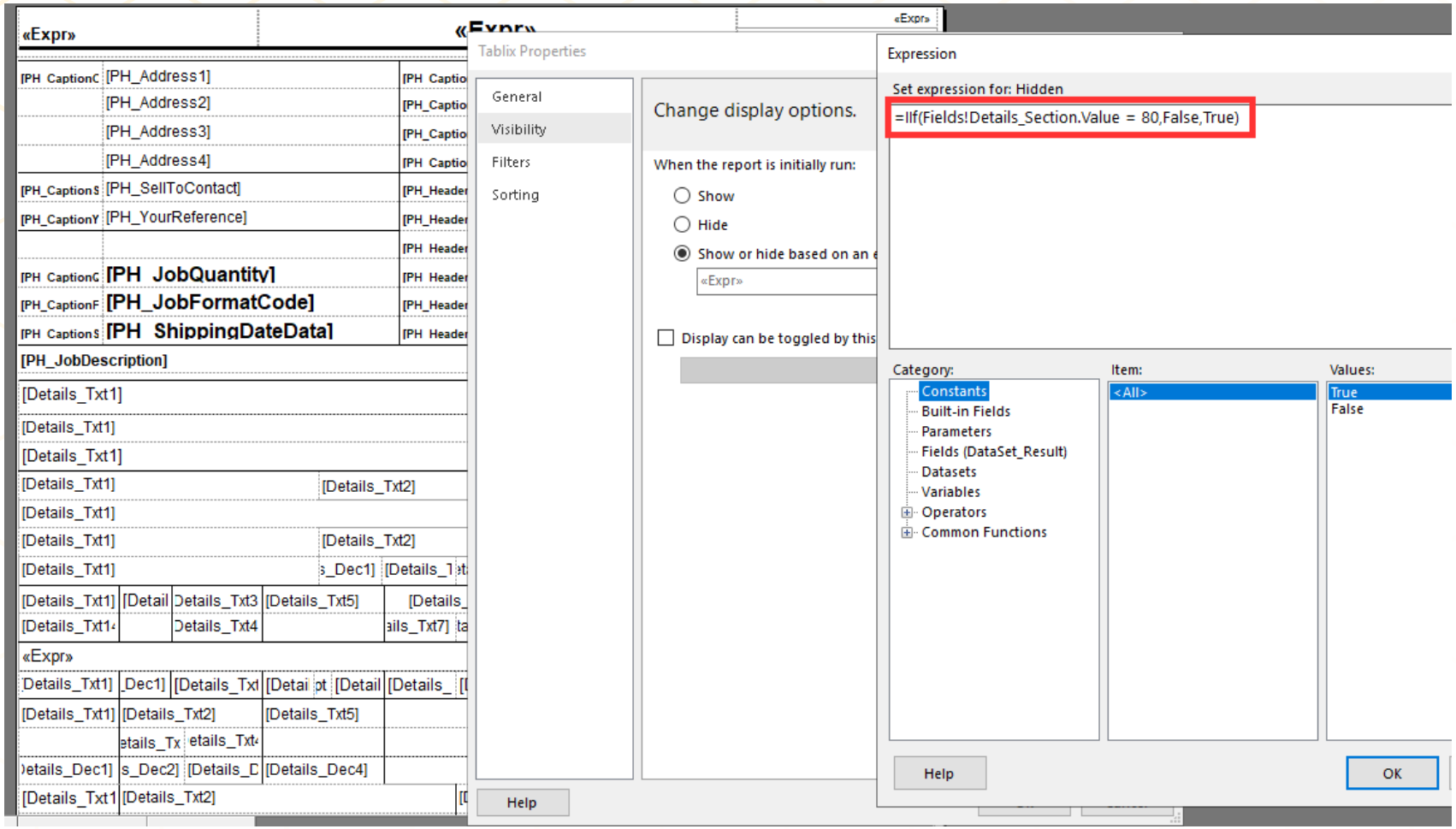
LineType:

```
 1 : Delimiter Line
 2 : Header (Department Name etc.)
 3 : Comment Text (Full Width)
 4 : Side by Side: Comment Text (Half Width) + UserFields
 5 : UserFields
 6 : Side by Side: Comment Text (Half Width) + UserFields  (Same as 4)
 7 : UserFields                                            (Same as 5)
20 : Calc. Unit Details
50 : Press - Process Head
51 : Press - Process Subject Body
52 : Press - Process Body
55 : Press - Plate Change Head
57 : Press - Plate Change Body
60 : Postpress - Process Head
61 : Postpress - Process Subject Body
62 : Postpress - Process Body
70 : Press Continuous - Process Head
71 : Press Continuous - Process Subject Body
72 : Press Continuous - Process Body
80 : Shipments
```

Workshop JobTicket 3/8

Job Ticket

Layout visibility expression for each section based on section value



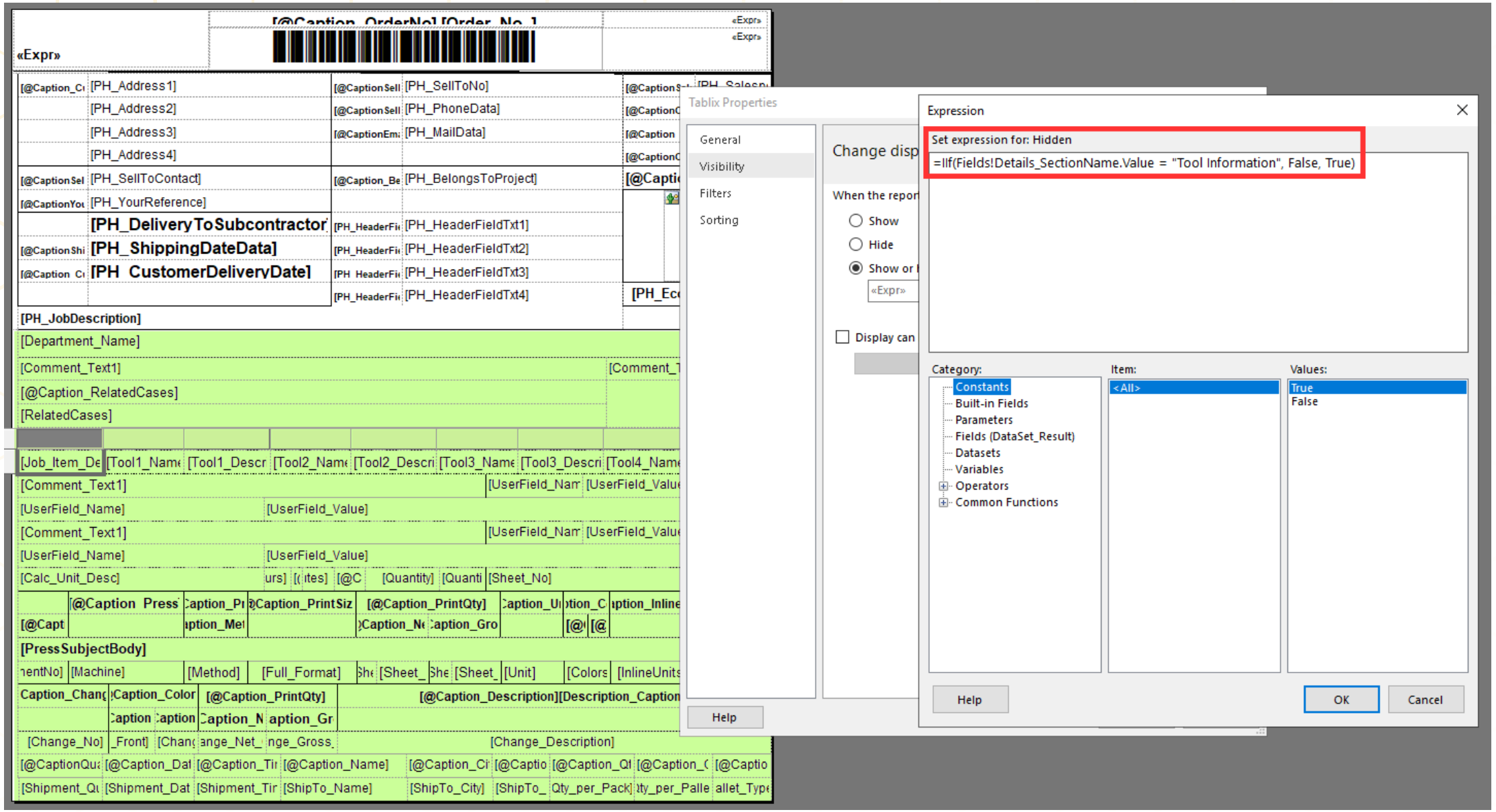
Workshop JobTicket 4/8

OLD Job ticket has 1 Event for customization

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS JobTicket Management", 'OnAfterGetReportBuffer', '',
true, true)]
    local procedure OnAfterGetReportBuffer(var in_JobRec: Record "PVS Job"; var in_VERSION_COUNTER: Integer;
var Out_BufferRecTmp: Record "PVS Sorting Buffer" temporary; var Result: Boolean)
    begin
        // the generated dataset is in Out_BufferRecTmp
        // this can now be manipulated
        end;
```

- Task 1
  - Change Job Item Description

Workshop JobTicket 5/8



Workshop JobTicket 6/8

OnBeforeCalculateQtyPerPlateChange
OnBeforeDepartmentHeader
OnBeforeDepartmentLines
OnBeforeFindJobShipments
OnBeforeFooterLines
OnBeforeGetPlateChanges
OnBeforeGetPressBody
OnBeforeGetPressSubject
OnBeforeHeaderLines
OnAfterGetPressBody
OnAfterGetJobTicket
OnAfterHeaderLines
OnAfterDepartmentLines
OnAfterFooterLines

Change the Press Subject Body example

```
    [EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS JobTicket Management", 'OnBeforeGetPressSubject',
'', true, true)]
    local procedure OnBeforeGetPressSubject(var in_JobItem: Record "PVS Job Item"; in_DepartmentRec: Record
"PVS Department"; var PressSubjectTemp: Record "PVS Job Ticket Temp" temporary; var lastentry: Integer; var
isHandled: Boolean)

    begin
        PressSubjectTemp.PressSubjectBody := in_JobItem.Description;
        PressSubjectTemp."Report Section" := "PVS LineTypes"::"Press - Process Subject Body";
        LastEntry += 1;
        PressSubjectTemp.PK1_Integer1 := LastEntry;
        PressSubjectTemp.KEY_Sorting_Decimal1 := LastEntry * 1000;
        PressSubjectTemp.Insert();
        isHandled := true;
    end;
```

Workshop JobTicket 7/8

Add extra line under each printing process

- Create new line after the PressBody with a new key – use PK1_Integer2
- Set a new report section and a new text field to be printed
- Add new section in the layout

```
[EventSubscriber(ObjectType::Codeunit, Codeunit::"PVS JobTicket Management", 'OnAfterGetPressBody', '', true, true)]
    local procedure OnAfterGetPressBody(var in_SheetRec: Record "PVS Job Sheet"; in_DepartmentRec: Record "PVS Department";
    var PressBodyTemp: Record "PVS Job Ticket Temp" temporary; var lastentry: Integer)
    begin
            //Logic
    end;

    procedure GetJobTicketTemp(var JobTicketTemp: Record "PVS Job Ticket Temp")
    begin
        JobTicketTemp := Global_JobTicketTemp;
    end;

    procedure SetJobTicketTemp(var JobTicketTemp: Record "PVS Job Ticket Temp")
    begin
        Global_JobTicketTemp := JobTicketTemp;
    end;

    var
        Global_JobTicketTemp: Record "PVS Job Ticket Temp";
```

Workshop JobTicket 8/8

```
tableextension 50100 "PTE Job Ticket" extends "PVS Job Ticket Temp“
{
        field(50100; CustomTextField; Text[50])
enumextension 50101 "PTE Line Types" extends "PVS LineTypes"
{
    value(50100; MyNewSection)
    {
        Caption = 'My new section', Locked = true;
reportextension 50101 "PTE Job Ticket 2" extends "PVS Job Ticket 2"
{    dataset
    {
        add(LINE_DETAILS)
        {
            column(CustomTextField; JobTicketTempS.CustomTextField){}
        }
        modify(LINE_DETAILS)
        {
            trigger OnAfterAfterGetRecord()
            var
                PTEJobTicket: Codeunit "PTE Job Ticket";
            begin
                PTEJobTicket.GetJobTicketTemp(JobTicketTempS);
            end;
        }
    }
    var
        JobTicketTempS: Record "PVS Job Ticket Temp";
}
```

Workshop KPI 1/2

KPI Calculations

Possible to calculate/update 10 KPI fields on PVS Job
KPI's can be set on different product groups

Example:

```
[EventSubscriber(ObjectType::Table, Database::"PVS Job", 'OnAfterValidate_Price_Method', '', true,
false)]
procedure OnAfterValidate_Price_Method(var in_Rec: Record "PVS Job");
var
KPI_Value1, KPI_Value2, KPI_Value3: Decimal;
begin
// If you only want to calculate for a certain prod group you can add this check
if in_Rec."Product Group" <> 'POSTER' then
exit;
// Calculate up to 10 KPI values somehow
KPI_Value1 := 13;
KPI_Value2 := 23;
KPI_Value3 := KPI_Value1 + KPI_Value2;
// Assign the KPI values to the Job
in_Rec."Misc. 1" := KPI_Value1;
in_Rec."Misc. 2" := KPI_Value2;
in_Rec."Misc. 3" := KPI_Value3;
end;
```

Workshop Unit Conversion

In the printing industry, many different units are used.
For example, paper could be handled in weight *and* pieces.
On the item the consultant/user has entered known information about the item's dimension, and with help of that we have standard functions to transform/convert from different units.

PrintVis Standard Units

```
PVSItemManagement.Item_InventoryUnit_Factor(Item,FromUnit,ToUnit) = factor
```

Workshop KPI 2/2

KPI Calculations

- Task 1
  - Create KPI field that displays cost of all plates in the estimate connected to the job
    Use CacheManagement.READ_Tmp_Job_CalcUnitDetails to get the Calc Unit Details
    Filter on Item Type
- Task 2
  - Create KPI field that displays the price per 1000pcs for the selected Paper
  - Convert the value of previous KPI from pcs to Weight
  - Convert the value of previous KPI from pcs to Area
  - Add the KPI fields next to the quoted price on the job list part on the case card.

PRINTVIS

Thank You

See You Next Time!