

Building a Product Table with Inventory Sync in Pega Constellation: A POC Walkthrough

As Constellation becomes the new standard for building modern Pega applications, many of us are exploring how to implement common business use cases using its new design system and capabilities. In this article, We'll Walk you through a **proof of concept (POC)** We built using Constellation that demonstrates how to create a **dynamic product table** with **real-time calculations** and **inventory management**.

Use Case Overview

The goal was to build a **product selection interface** where users can:

- Select products from a predefined list (sourced from a data type).
- Enter quantities for each product.
- Automatically calculate the **product total** (price × quantity).
- Display a **Total Amount** (sum of all product totals).
- Update the **inventory** by subtracting selected quantities.

Data Model

Edit Property: List Of Products [Available]

CL: SL-TellUsMore-Work-Incident ID: ListOfProducts RS: TellUsMore:01-01-04

General

Advanced

Specifications

History

Property type

Page List

Page definition *
SL-TellUsMore-Data-Products

Data access

☐ Manual

☐ Refer to a data page

☒ Copy data from a data page

When the parameter values being passed to the data page change and a non-parameter value is referenced, a new data page is loaded and copied into this property.

Interactions with the property happen on the copied data.

☐ Load each page in this page list individually

Data Page: *
D_ProductsList

SL-TellUsMore-Data-Products

Parameters

No parameters to set.

Optional Data Mapping:

Parameters

No parameters to set.

Data Type: Products

SL-TellUsMore-Data-Products RS: TellUsMore

Actions: Save

Data Model

Usage

Sources

Records

UX

Actions

Test cases

Settings

Search...

Show system fields

View data model

Primary Fields

Add Field

Name	ID	Type	Options	Application Layer
Globally unique ID	pyGUID	Text (single line)	Key (autogenerated); Read-only	Pega Platform
Label	pyLabel	Text (single line)		Pega Platform
NoOfSelectedProduct	NoOfSelectedProduct	Integer		Tell Us More
Product Name	ProductName	Text (single line)		Tell Us More
Product Price	ProductPrice	Currency		Tell Us More
Product Quantity	ProductQuantity	Integer		Tell Us More
Product Total	ProductTotal	Currency	Calculated; Read-only	Tell Us More

I created a **Product** data type with the following fields:

- ProductName (Text)
- ProductPrice (Decimal)
- ProductQuantity (Integer)
- ProductTotal (Read-only, calculated: ProductPrice * ProductQuantity)

The **Total Amount** is a read-only field calculated as the **sum of all ProductTotal values**.

UX Configuration in Constellation

On the step in the case type:

- I used an **embedded list** of the Product data type.

The screenshot shows the 'Edit View Determine Category' configuration window in Constellation. On the left, a 'List Of Products' embedded list is visible, containing columns for 'Label', 'Product Name', 'Product Price', 'Product Quantity', and 'Product Total'. Below the list is a 'Total Amount' field. On the right, the configuration panel includes a 'Determine Category' dropdown, a 'Label' dropdown, and a 'Instructions' dropdown. The 'Instructions' dropdown is set to 'Override case data instructions'. Below these are 'Fields' for 'List Of Products' (Quantity) and 'Total Amount' (Currency). The 'Advanced' section is expanded.

The screenshot shows the 'Edit Property: List Of Products [Available]' configuration window in Constellation. The 'Property type' section is expanded, showing the 'Page List' configuration. The 'Page definition' is set to 'SL-TellUsMore-Data-Products'. The 'Data access' section is expanded, showing the 'Data Page' set to 'D_ProductsList'. The 'Parameters' section is expanded, showing 'No parameters to set.'.

- The table is **editable**, allowing users to input quantities.

Constellation

List Of Products

Label	Product Name	Product Price	Product Total	NoOfSelectedProduct
No records found.				

Cancel

Submits

Edit field: List Of Products

Display as: Table

Add/edit records in: Table rows

Columns

- Label
- Product Name
- Product Price
- Primary fields

Add label: Default

Column to take up remaining width: Default

Row density: Default

Conditions

Allow adding: Never

Allow editing: Always

Allow deleting: Never

Allow row re-ordering: Never

Record-level conditions

Allow deleting records: Never

Display Conditions Pre/Post Processing

Previous as: Constellation

List Of Products

Label	Product Name	Product Price	Product Total	NoOfSelectedProduct
No records found.				

Cancel

Submits

Configure field: Product Name

Field label: Custom

Custom field label: Product Name

Edit mode: Read-only

Column settings

Column width: Auto

Conditions

Visibility: Always

Advanced

Test ID

Cancel

Save

Conditions

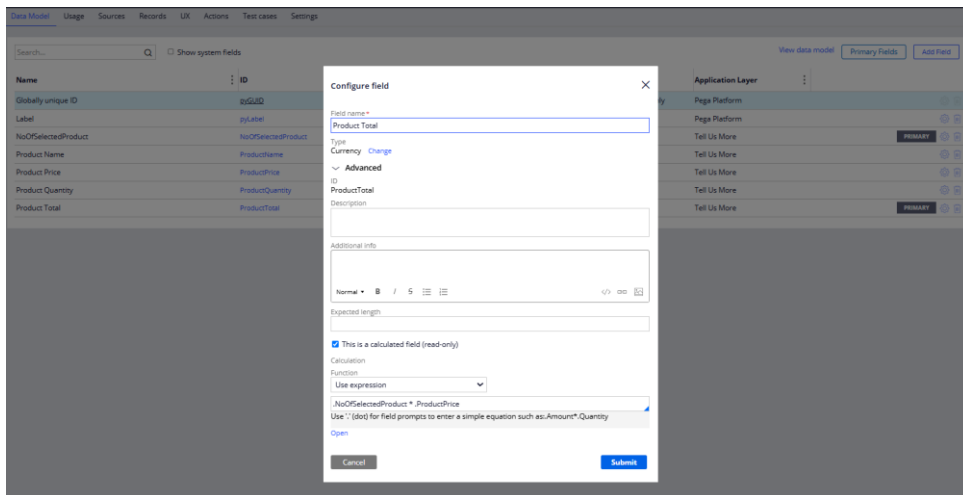
Allow adding: Never

Allow editing: Always

Allow deleting: Never

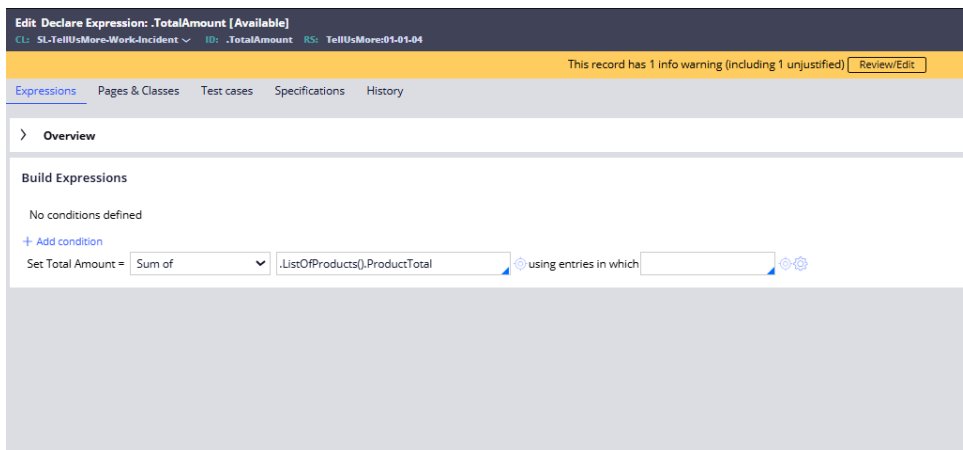
Allow row re-ordering: Never

- The ProductTotal field is **auto-calculated** and read-only.



- The TotalAmount field is displayed below the table and updates in real time.

Declare Expression Field



Inventory Management Logic

- To manage inventory:
- When a product is added or its quantity is updated, the system **subtracts the quantity** from the available inventory.

Data Model

Usage

Sources

Records

UX

Actions

Test cases

Settings

Source

Local data storage

Actions

Export

Import

Search...

Label	Product Name	Product Price	Product Quantity	Product Total	NoOfSelectedProduct	Delete
Chips	Uncle Chips	\$10.00	75	0.000000000	0	
Milk	Amul Milk	\$30.00	65	0.000000000	0	
Biscuit	Britania Biscuits	\$30.00	55	0.000000000	0	

+ Add record

I added **validation** to prevent users from entering quantities greater than what's available.

Create Incident (I-4024)

Determine Incident Type

Product Quantity Less Than Product In Added Cart

Please select the type and subtype of your incident

List Of Products

Label	Product Name	Product Price	Product Total	NoOfSelectedProduct
Chips	Uncle Chips	\$10.00	\$1,000.00	100
Milk	Amul Milk	\$30.00	\$0.00	0
Biscuit	Britania Biscuits	\$30.00	\$0.00	0

Total Amount
\$1,000.00

CancelNext

Added valid Number of items

Create Incident (I-4025)

Determine Incident Type

Please select the type and subtype of your incident

List Of Products

Label	Product Name	Product Price	Product Total	NoOfSelectedProduct
Chips	Uncle Chips	\$10.00	\$0.00	10
Milk	Amul Milk	\$30.00	\$0.00	0
Biscuit	Britania Biscuits	\$30.00	\$0.00	0

Total Amount
\$0.00

CancelNext

Data Type: Products

SL-TestUsMore-Data-Products

RS TestUsMore

Data Model

Usage

Sources

Records

LUX

Actions

Test cases

Settings

Source

Local data storage

Actions

Export

Import

Search...

Label	Product Name	Product Price	Product Quantity	Product Total	NoOfSelectedProduct	Delete
Chips	Uncle Chips	\$10.00	65	0	0	
Milk	Amul Milk	\$30.00	65	0	0	
Biscuit	Britania Biscuits	\$30.00	55	0.000000000	0	

+ Add record

- If a product is removed or quantity is reduced, the inventory is **restored accordingly**.
- **UpdateInventory** : Activity is called at post processing of the flow action

Post-processing

Occurs after validation passes and this action is submitted

Run robotic automation

Description

Savable data pages

No items

+ Add data page

Apply cost

Apply data transform

Refresh parameters

Run activity

UpdateInventory

Refresh parameters

Parameters

ProductQuantity

NoOfSelectedItem

Back-to-back processing configuration

☒ Look for an assignment to perform

Edit Activity: UpdateInventory (Available)

SL-TestUsMore-Work-Inventory

UpdateInventory

TestUsMore-01-01-04

Delete

Actions

Save

This record has 4 severe or moderate warnings (including 4 unjustified) and 2 info warnings

Review/Edit

Steps

Parameters

Pages & Classes

Security

Test cases

Specifications

History

Label	Method	Step page	Description
1. Loop When	Page-Clear-Messages	pyWorkPage	Clear all messages on the step page
2. Loop When	ListOfProducts		
1. Loop When	Page-Set-Messages		Add a message to the page

Method Parameters

Page

pyWorkPage

Category

Message

Product Quantity Less Than

2. Loop When	Page-Copy		Copy page
3. Loop When	Property-Set	TempListOfProducts	Set property values

Method Parameters

PropertiesName	PropertiesValue
Param_ProductQuantity	ProductQuantity
Param_NoOfSelectedItem	NoOfSelectedProduct
ProductQuantity	Param_ProductQuantity-Param_NoOfSelectedItem
NoOfSelectedProduct	0

4. Loop When

Obj-Save

TempListOfProducts

Save record to the database

3. Loop When

Commit

TempListOfProducts

Commit changes to database

Method Parameters

No parameters

+ Add a step

Collapse all steps

Key Learnings

- **Constellation's editable tables** are powerful for building interactive, real-time interfaces.
- **Declarative expressions** make it easy to calculate totals without writing custom code.
- **Inventory sync** can be handled using data transforms or decision rules triggered on change.
- **Validation rules** are essential to ensure data integrity and a smooth user experience.

This POC helped me understand how to use Constellation's new capabilities to build a real-world use case with dynamic data, calculations, and backend logic. I hope this helps others who are just getting started with Constellation!

Feel free to reach out or comment if you'd like to see a demo or need help implementing something similar.