Report & signature sample

Outline

This document provides description how a generated Report and a Signature pad can be displayed side by side. Once signature is provided how report can be re-generated with a little help from JavaScript. The idea is that the user should see generated report before it is signed.

To achieve this, we'll put two media tabs side by side on a form. One will be configured to show a report file, another one to capture signature.

On a custom command on a form, the functionality will be activated, and a report will be generated from JavaScript. The first media tab, configured to show the report, will be reloaded to immediately show the report to the user.

Next, when the user provides a signature, the command can be activated again to re-generate the PDF and this time contain also signature.

Prerequisites

The sample is created for standard Product object ('sf_product2') in Salesforce. Of course, it can be adjusted for any other object.

1) The corresponding HTML file and JSBridge.js have to added to Offline HTML section. The HTML file then has to be added as Iframe on Product form.
2) Also, custom command to activate the functionality has to be added. (See details in Entity Form)
3) On the Product form, two media tabs have to be added. They need to have the exact names and have to be setup according to instructions in section Entity Form.
4) A report has to be created for Product object which displays the product data and the signature according to instructions in section Report Editor.

Notes

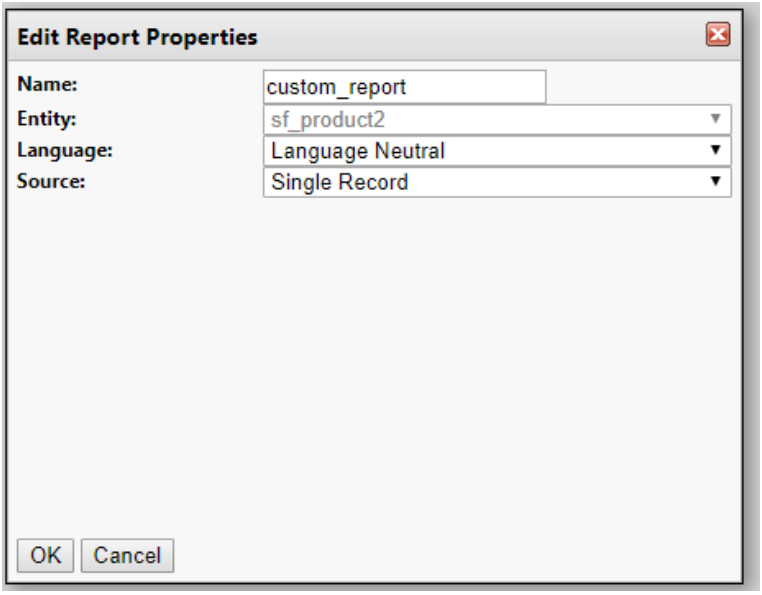
Here are listed implementation notes & considerations:

- The attached HTML contains constants at the beginning of the script that can be used to adjust the object this code is attached to, names of the report, file …
- Please note that user has to save the form in order to generate the report and similarly save the form once the signature is provided. The reason is that the report gets the data from database and therefore everything needs to be saved first.
- The JavaScript code works as follows. It generates the report from JavaScript code and stores it as Attachment with specified name. The media tab is configured to show Attachment with this name, so it reloads the media tab once the report is generated.
- The generated report will be stored as Attachment object. This is due to limitation in our JavaScript API, but if storing the report in Content Document is required, we can push for an extension of the JavaScript API.

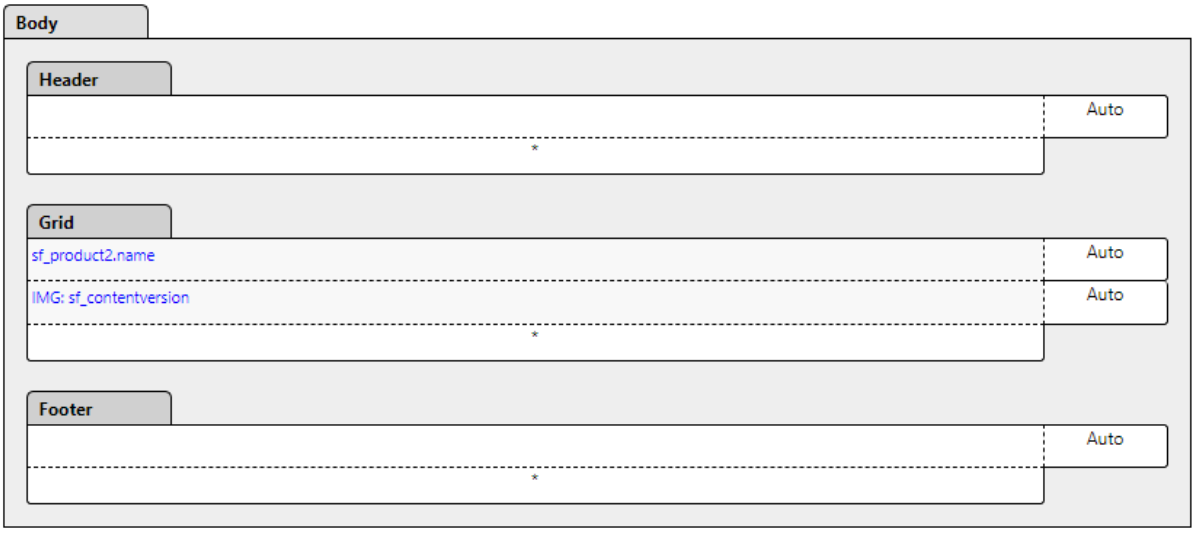
Report Editor

The first step is to prepare a Mobile Report with a Signature. Login to Resco console (salesforce.resco.net) and open Mobile Report Editor. There, create a new report based on following instructions.

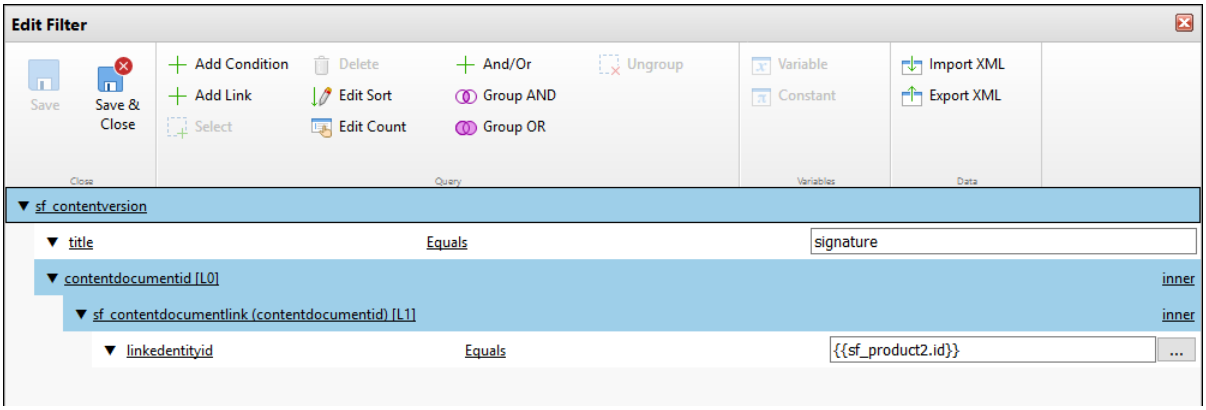
We've created a Report for Product object ('sf_product2') and for Single Record only (Source property). The name of the report we'll use 'custom_report'. Please note that if you want to use different name, you'll have to also adjust the JavaScript code in the HTML file.



Our report is quite simple, it contains one field from Product (name) and one Image with signature.



The signature is stored in Content Version (that is how we'll configure the Media Tab), thus you have to Add Image (on the top ribbon) and edit binding. This will be the query which retrieves the signature image, so update it as in the following picture:



Again, if you'll use a different object or a different signature title, this has to be reflected in the Woodford setup as well as JavaScript code.

Our example report can be found here:

custom_report.xml

Please don't forget that you need to Activate your report so that it can be visible in mobile app.

Woodford

Login to Resco console (salesforce.resco.net) and open Woodford. Our example uses Product, but you can choose a different object. However, that would require changes in the code and setup, so for the test of the functionality you should follow this tutorial as is.

OfflineHTML

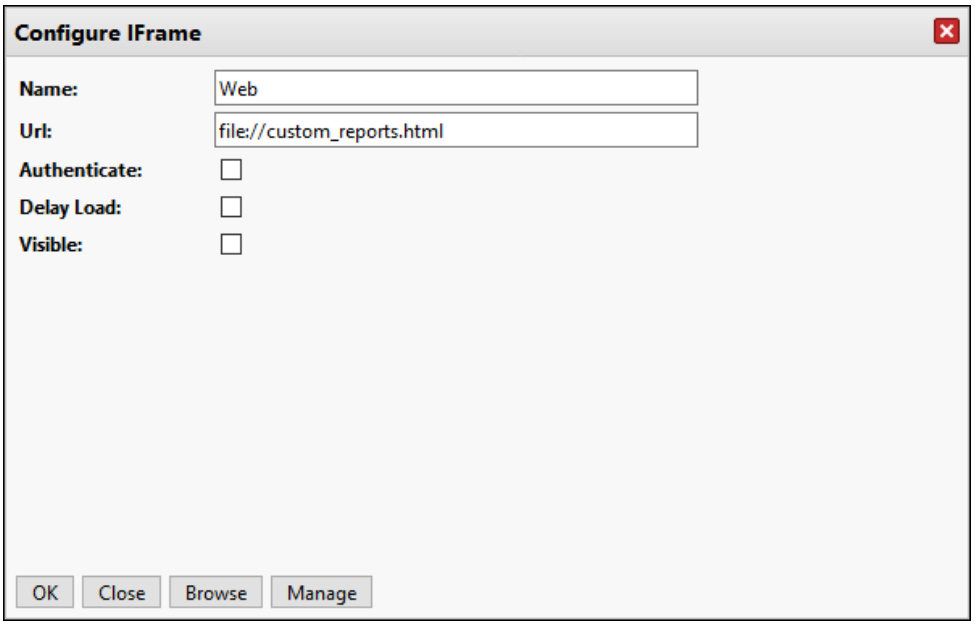
In Woodford you'll need to add the HTML file containing the script and Resco's latest JSBridge.js file. You can add them into the root, or if you already have some files there you can create a folder.

custom_reports.html

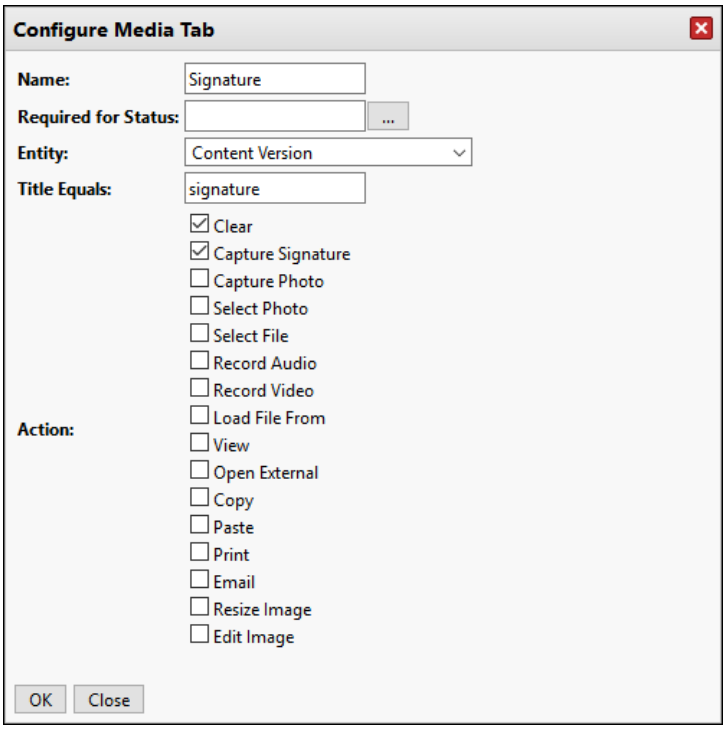
Entity Form

Open the Product form. At first, we'll create the custom command. In the top ribbon menu, in Commands section, click on Edit. Command dialog should appear. Click on New Command. Fill in the name and label. In the sample we're using the name of the command "custom_run_report". If you'll use a different name, change its name in the script (constant commandName). Save everything.

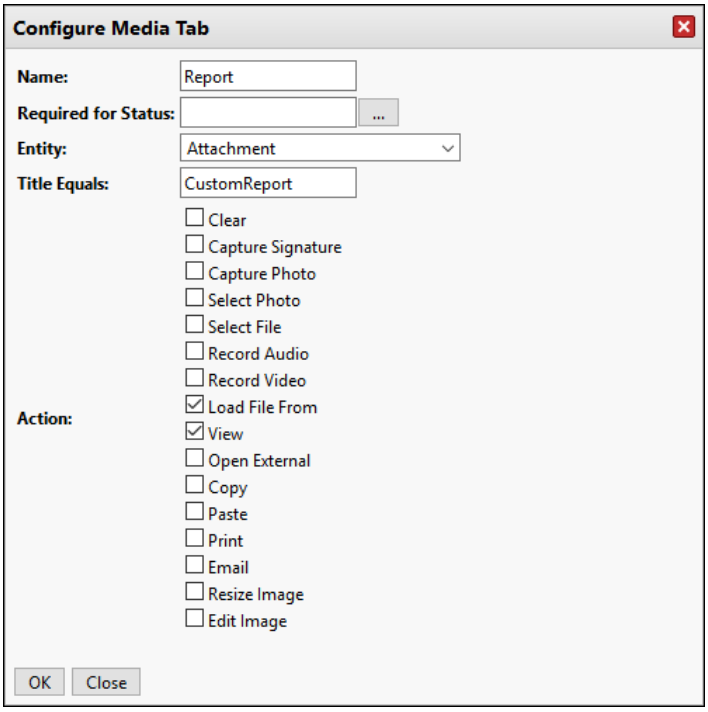
Second step is to add one iFrame with custom_report.html (Click Browse button to select the file from Offline HTML folder). It should be invisible (deactivate Visible and Delay Load):



Next, add a media tab for Signature, configure it as in the picture. Please note "Title Equals" must have the same value as "title" in the condition for Image in the report editor (so in our case it will be 'signature'):



Now add a media tab for the Report, and configure it as in the picture:

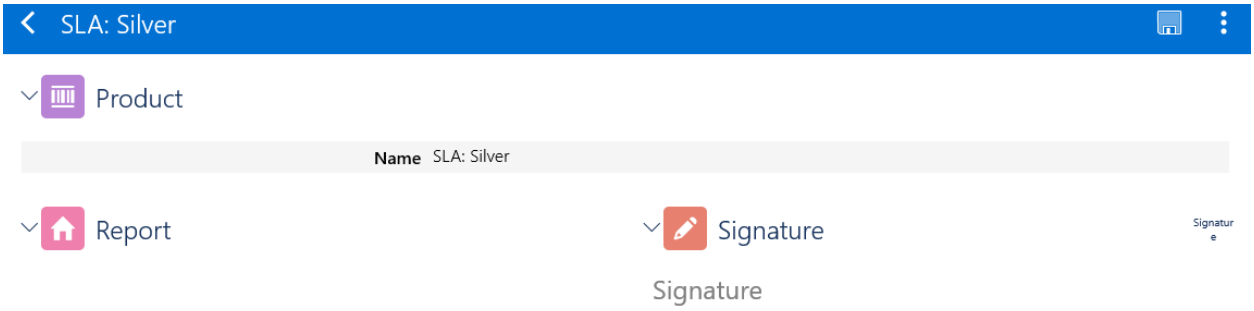


Again, the title has to match the title used in JavaScript code.

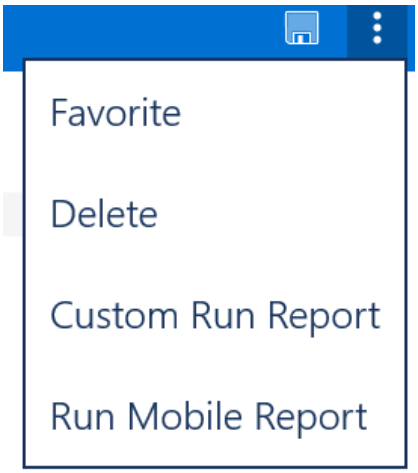
With all this set up, the sample should be ready.

Application walk-though

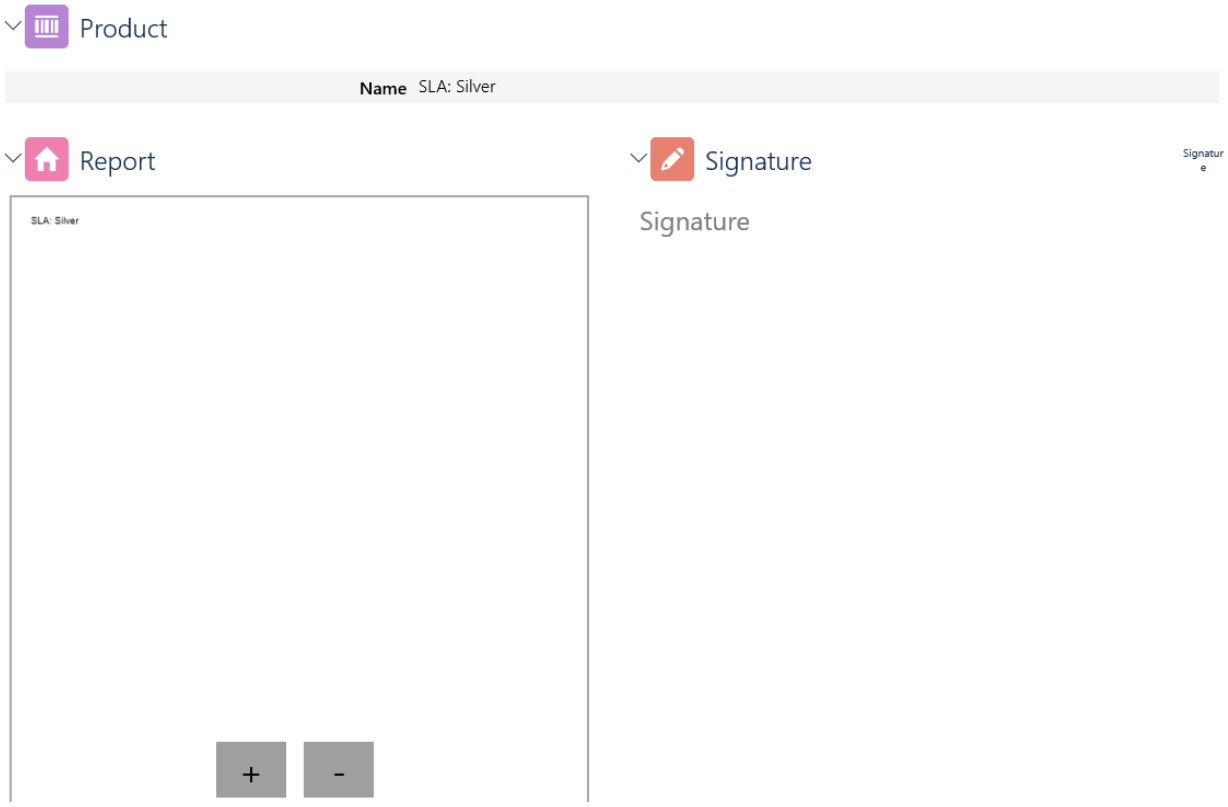
With everything correctly set up, the Product form in application should look something like this after synchronization.



In the top right corner, there is commands menu (three dots). One of the actions there is ‘Custom Run Report’ – that is the one we’ve added.

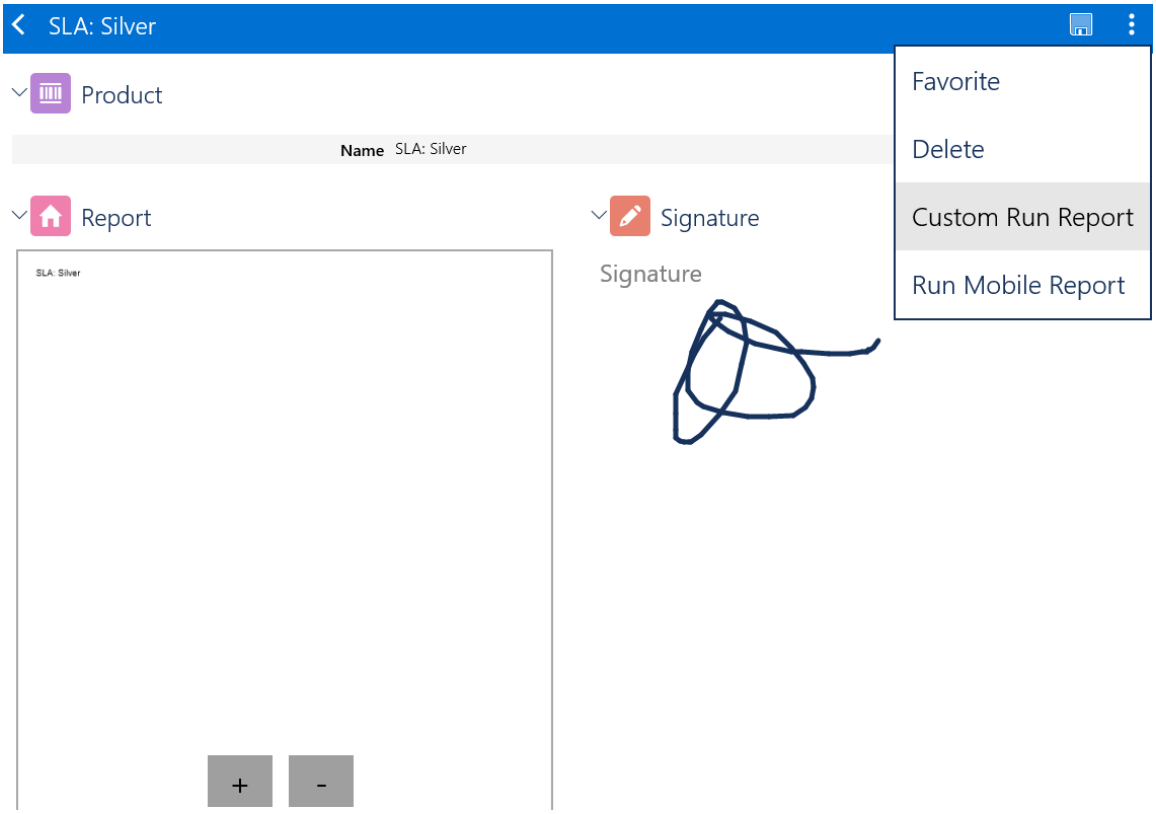


Clicking this command will run our JavaScript code. The JavaScript will generate the report and store it as Attachment. Then, it reloads the Media Tab. So, the result should be this.

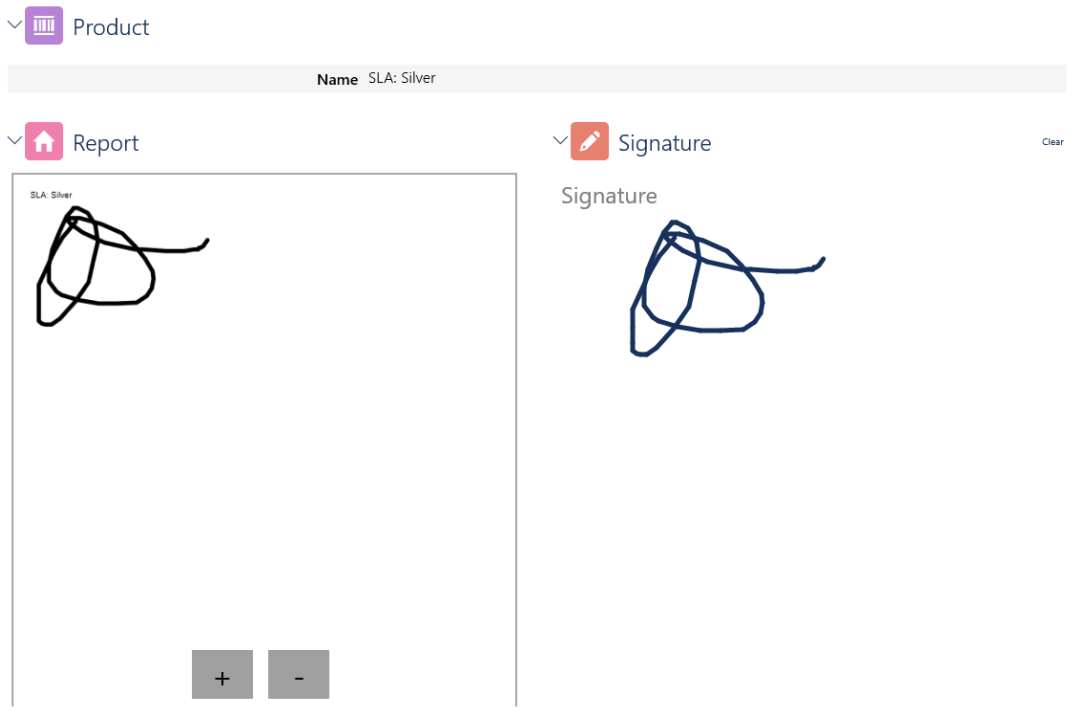


It might not look like much, but please remember that our report is very simple, it contains only the name and signature. Since the signature has not been provided, we see only the Product name, but good news is our report has been generated and is displayed in the media tab.

User might now inspect the report and decides to sign it. For this, he'll use the signature pad next to the report. Once the signature is provided the user has to SAVE the form and then can activate the custom command again ('Custom Run Report' from the top right three dots menu).



One the command is activated; the report is re-generated and looks like this.



Final notes

This sample demonstrates the main principle how the report experience can incorporate the signature more organically so that the user can see what document is being signed.

Of course, it can be extended and expanded further, e.g. the two media tabs can be only visible when the command is activated (and perhaps all the other tabs can be made invisible) to make the process even more straightforward. Or a button (link item) can be added to re-generate the report without the need to run the custom command again.