Retail Capital Streamlines Its Agreement Process with DocuSign for Salesforce and Custom Buttons

> "I know DocuSign works! If Salesforce themselves use it, it's gotta work."
> **- Steve Holyoake, Executive Head: Systems and Risk**

Small and medium enterprises (SMEs) contribute to more than 65% of South Africa's employment and 50% of the country's GDP. However, the majority of those businesses indicate lack of access to working capital as the number one inhibitor to growth. That's where Retail Capital comes in. Retail Capital is a financial services company in South Africa that provides SMEs with the capital they need through partnerships and funding.

The financing provided by Retail Capital is not a loan, but instead advanced for a fixed fee with no interest. Retail Capital's team are experts in analyzing business turnover and therefore do the critical analysis of risk up-front. They only provide capital to companies with an acceptable level of risk and monitor them accordingly.

Retail Capital uses Salesforce to manage its business. Naturally, Retail Capital looked for the tight integration made possible by DocuSign for Salesforce. In fact, part of the decision to go with DocuSign was, as Steve Holyoake (Retail Capital's Executive Head of Systems) indicated, "I know DocuSign works! If Salesforce themselves use it, it's gotta work."

Humble Beginnings

Retail Capital embarked on a journey to shorten the time to agreement and reduce costs in their agreement process. Prior to implementing DocuSign, they would use details captured in Salesforce as merge fields in a template to create contracts, but produced in hard copy form. An account manager at Retail Capital would then courier the contract or physically travel to a customer (generically referred to as a *merchant*) site. Quite often, the contracts would be returned to Retail Capital with insufficient signatures, the wrong people signing the contracts, or the signatures or other data elements would be in the wrong place on the document. In such cases, the process starts over again, resulting in additional time and costs. Since implementing DocuSign for Salesforce, none of these troublesome situations are a factor.

Despite the success of Retail Capital's DocuSign for Salesforce implementation, they didn't actually begin this journey with DocuSign. They started with a DocuSign competitor and managed to get some functionality working, but failed to achieve overall success with that product.

The next step was to implement DocuSign. While the devil is always in the details, Retail Capital's business requirements were not too complex. To begin, they only wanted to test the process for repeat customers, as these contracts are relatively simple (compared to those for first time customers) being a single standard agreement for signature that requires these three separate signing roles in this order:

1. **Checker** – The Retail Capital account representative that is responsible for verifying all data on the agreement, including terms, names, addresses, etc.
2. **Merchant Recipients** – The merchant signature that is authorized to enter into the agreement on behalf of the customer/client. In fact, there may be multiple signatures required on behalf of the merchant, depending on the legal structure and number of shareholders. This required Retail Capital to create separate DocuSign templates to cater for different numbers of merchant recipients.
3. **Retail Capital Signing Groups** – A group of authorized signers on behalf to Retail Capital to bind the agreement, any one of which can sign.

> "What used to take weeks for a contract to be signed, plus the cost of a courier or an account manager's visit to the merchant, now averages less than one hour to complete."
>
> **- Steve Holyoake, Executive Head: Systems and Risk**

When a Developer is Not a Developer

Who says you need to be a software developer (or a trained IT professional for that matter) to develop with DocuSign technologies? Certainly not Retail Capital! Steve Holyoake is the Executive Head of Systems and Risk at Retail Capital, but is not an IT guy, per se. In other words, he does not write code for a living. However, that didn't stop him from developing his solution using DocuSign for Salesforce and custom buttons. Custom buttons are a way to invoke custom actions or JavaScript code from within the Salesforce environment.

He started by analyzing the business process needed with the three signing roles. Then Steve decided to write the code himself by using the DocuSign for Salesforce Administration Guide and reaching out to DocuSign Support to answer any additional questions he had.

How it Works

The Retail Capital agreement process starts on a custom Salesforce object called **Applications**, (where the 'deal' is underwritten) and is shown in Figure 1. It displays data for a specific application and contains a custom button named **Re-advance E-contract**.

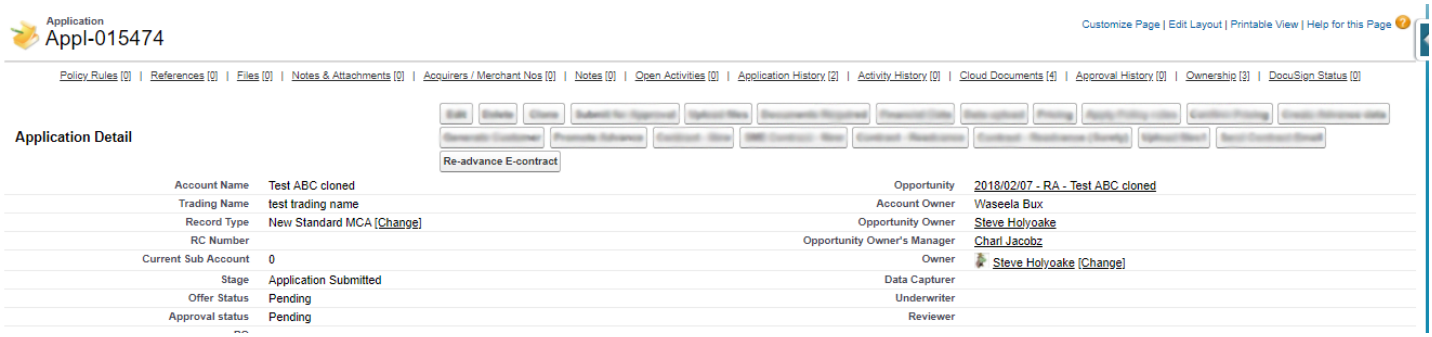

Figure 1: Salesforce custom Applications object with a custom button.

Clicking the **Re-advance E-contract** custom button calls JavaScript code, as shown in Figure 2, that controls the process and selects the correct DocuSign template.

```
OnClick JavaScript    {!REQUIRESCRIPT("/apex/dsfs__DocuSign_JavaScript")}

{!requireScript("/soap/ajax/35.0/connection.js")}

var RC = "";var RSL="";var RSRO="";var RROS="";var CCRM="";var CCTM="";var CCNM="";var CRCL="; var CRL="";var OCO="";var DST="";var LA="";var CEM="";var CES="";var STB="";var SSB="";var SES="";var SEM="";var SRS="";var SCS ="";var RES="";

var result = sforce.connection.query("Select Ownership__c.Contact_Name_text_value__c, Ownership__c.e_mail__c from Ownership__c where Application__r.Name = '" + "{!Application__c.Name}'");

var contactRole1 = result.getArray('records')[0];
var contactRole2 = result.getArray('records')[1];
var contactRole3 = result.getArray('records')[2];

var numown = "{!Application__c.Count_of_Owner_records__c}"


if(numown == 3){CRL="Email~{!User.Email};LastName~{!User.LastName};Role~Checker;RoutingOrder~1,Email~"+contactRole1.e_mail__c+";LastName~"+contactRole1.Contact_Name_text_value__c+";Role~Signer 1;RoutingOrder~2,Email~"+contactRole2.e_mail__c+";LastName~"+contactRole2.Contact_Name_text_value__c+";Role~Signer 2;RoutingOrder~2,Email~"+contactRole3.e_mail__c+";LastName~"+contactRole3.Contact_Name_text_value__c+";Role~Signer 3;RoutingOrder~2";}

else if(numown == 2){CRL="Email~{!User.Email};LastName~{!User.LastName};Role~Checker;RoutingOrder~1,Email~"+contactRole1.e_mail__c+";LastName~"+contactRole1.Contact_Name_text_value__c+";Role~Signer 1;RoutingOrder~2,Email~"+contactRole2.e_mail__c+";LastName~"+contactRole2.Contact_Name_text_value__c+";Role~Signer 2;RoutingOrder~2";}

else if(numown == 1){CRL="Email~{!User.Email};LastName~{!User.LastName};Role~Checker;RoutingOrder~1,Email~"+contactRole1.e_mail__c+";LastName~"+contactRole1.Contact_Name_text_value__c+";Role~Signer 1;RoutingOrder~2";}

CCRM='Checker~Checker;Signer 1~Signer1;Signer 2~Signer2;Signer 3~Signer3'


if(numown == 3){DST='[redacted]';}
else if(numown == 2){DST='[redacted]';}
else if(numown == 1){DST='[redacted]';}

window.location.href = "/apex/dsfs__DocuSign_CreateEnvelope?DSEID=0&SourceID={!Application__c.Id}&DST="+DST+"&CRCL="+CRCL+"&CRL="+CRL+"&OCO="+OCO;
```

Figure 2: JavaScript code for the custom button that invokes the DocuSign process.

By following DocuSign guidance on custom buttons, Steve wrote the JavaScript code to perform these actions:

- Pull contract data from Salesforce fields to populate the DocuSign template
- Select the correct DocuSign template, based on the number of merchant signers
- Determine correct routing order for the three signing roles
- Begin the electronic signing process using standard DocuSign for Salesforce workflow

You customize DocuSign for Salesforce with JavaScript code by working with predefined variables and objects. Steve followed this code example as a starting point, but then customized it for his needs. At a high level, Steve defined three variables to hold each of the required signing roles. He then determined how many owners there are at the merchant and assigned the **CRL** (custom recipient list) variable to the correct recipients in a specific signing order. The **CCRM** (custom contact role map) variable is used to map Salesforce roles to DocuSign template roles. Then Steve assigned the **DST** (DocuSign Template ID) to the hard-coded value of the template uploaded to the Retail Capital DocuSign account. He designed his templates to vary based on the number of merchant signers, so he applied that logic to the **DST** value. Finally, he called the **dsfs__DocuSign_CreateEnvelope** method to pass all the configuration variables and begin the process of creating the DocuSign envelope and routing to the signers.

DocuSign for Salesforce functionality manages the rest of the process. The next thing it does, as shown in Figure 3, is present the user with a screen showing the template selected, the recipients with the correct signing role and signing order, along with some additional details.

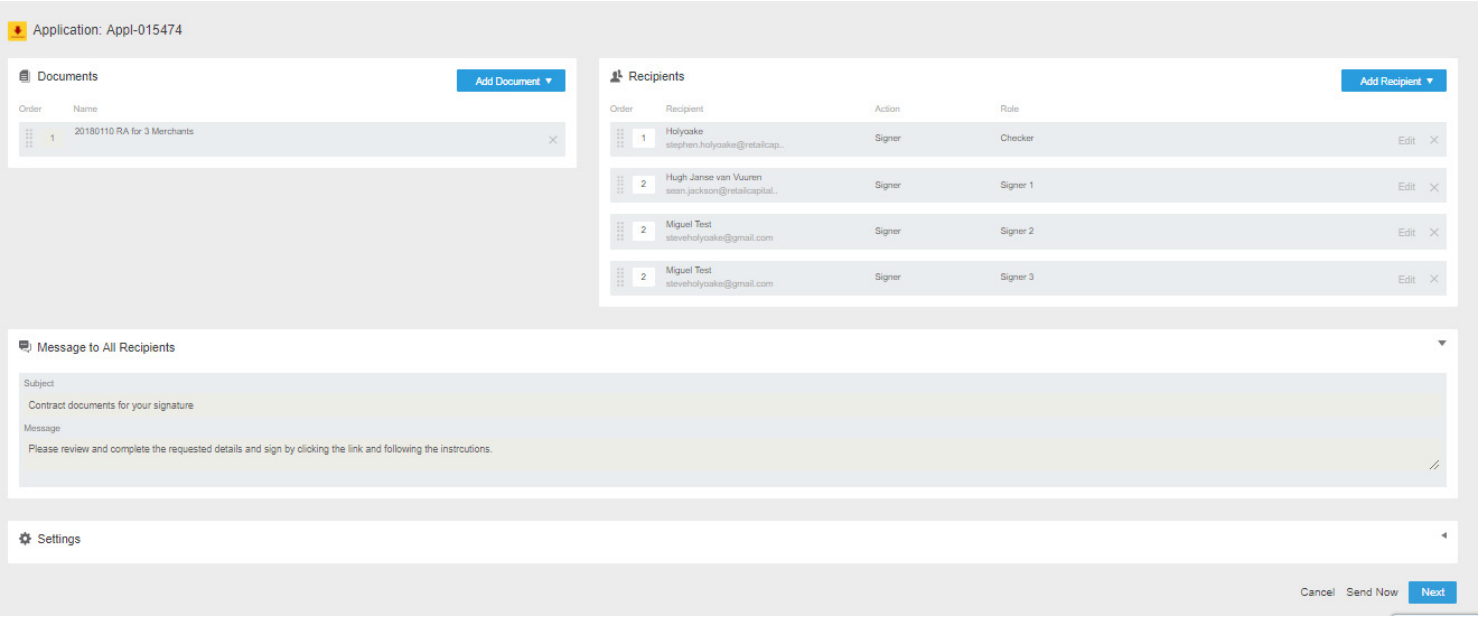

Figure 3: DocuSign for Salesforce Status.

When the user sends the contract, DocuSign routes it accordingly. The first stop is the checker at Retail Capital, who is required to verify every data element by clicking the corresponding **check** button. After verifying all data, the checker clicks **Finish**, as shown in Figure 4, and the contract is routed to the merchant.

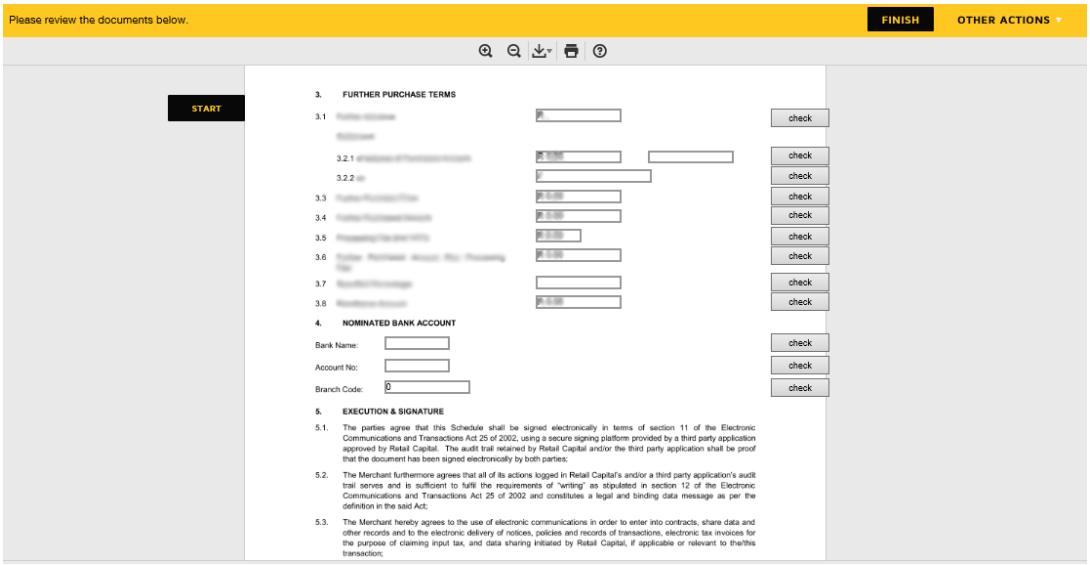

Figure 4: Contract routing, starting with the checker signing role.

Piloting and Capitalizing on the Future

As with most new business processes, Retail Capital piloted the custom DocuSign for Salesforce integration with a few key customers. They could begin with as many or as few as they like simply because the process is semi-automated, in that it starts by clicking the Salesforce custom button. If they want to pilot with fewer customers, they simply don't click the button until they are comfortable that the process works.

Retail Capital went live in February 2018, and in less than one month, they are already seeing major benefits. They send approximately 100 contracts per month to existing customers. It could take weeks for a signed contract to be received back by Retail Capital and, on average, it took several days. It also incurred the cost of a courier or an account manager's visit to the merchant. Now the whole process averages less than one hour to complete. With only one month elapsing since the integration went live, it is too soon to precisely quantify the return on investment (ROI). However, they certainly are saving money on the cost of couriers and other related expenses, plus it also means Retail Capital's customers are happy to receive funds earlier to grow their businesses.

Retail Capital plans to leverage their investment with additional DocuSign for Salesforce use-cases. This includes new customers, which requires additional documents to be signed – resulting in additional DocuSign templates and custom JavaScript logic. Steve will be happy to don his developer hat once more to achieve significant results for Retail Capital and DocuSign.

You can try the power and flexibility of the DocuSign for Salesforce or the DocuSign eSignature API for yourself with a free developer sandbox – just visit the DocuSign Developer Center, and see for yourself why Retail Capital chose DocuSign for Salesforce.

Follow Us:

About DocuSign

DocuSign® is changing how business gets done by empowering anyone to send, sign and manage documents anytime, anywhere, on any device with trust and confidence. DocuSign and Go to keep life and business moving forward.

For U.S. inquiries: toll free **866.219.4318** | **DocuSign.com**

For EMEA inquiries: phone **+44 203 714 4800** | email: **emea@docusign.com** | **docusign.co.uk**

For APAC inquiries: phone **+61 2 9392 1998** | email: **apac@docusign.com** | **docusign.com.au**

Copyright © 2003-2017 DocuSign, Inc. All rights reserved. DocuSign, the DocuSign logo, "The Global Standard for Digital Transaction Management", "Close it in the Cloud", SecureFields, Stick-eTabs, PowerForms, "The fastest way to get a signature", The No-Paper logo, Smart Envelopes, SmartNav, "DocuSign It!", "The World Works Better with DocuSign" and ForceFields are trademarks or registered trademarks of DocuSign, Inc. in the United States and or other countries. All other trademarks and registered trademarks are the property of their respective holders.