

GUIDE INSTALLATION ET INTEGRATION

Juillet 2021



Sommaire

1. About KSL for Salesforce®	4
1.1. Document and e-mails generation and delivery	4
1.2. Automatic production of documents and e-mail KSL, via Process Builder	5
1.3. Management of documents and e-mails created by KSL	6
1.4. Document and e-mail template design based on shared components	8
1.5. Technical flow diagram	8
2. Installing KSL Suite from Salesforce® AppExchange©	10
2.1. Technical prerequisites when installing the KSL plug-in	10
2.2. Installing KSL from Salesforce® AppExchange®	10
3. Configuring KSL for Salesforce®	12
3.1. User configuration	12
3.2. Configuring the communication between Salesforce® and KSL servers	14
4. Produce on demand a KSL document from a Salesforce® object	17
4.1. Associate a KSL Template Library with a Salesforce Object	17
4.2. Integrate a library of KSL models	18
4.3. Integrate an action button (push-button)	21
4.4. Other parameters: retention of transactional documents	24
5. Produce a KSL document from a Salesforce® automated process	26
5.1. Configure a KSL process	26
5.2. Execute the KSL process from the Salesforce® Process Builder	33
5.3. Control the execution log	36
6. Pre-configure the fields of the e-mail sending window	38
6.1. Objective	38
6.2. E-mail configuration	38
7. Validation workflow example	41
7.1. Creation of a queue	41
7.2. The validation workflow	41
8. Configuration of external document storage	42
8.1. General operation	43
8.2. Technical description	44
8.3. Implementing APEX classes	46
8.4. Setting permissions	51
8.5. Configuration for external storage	52
9. Annexes	53
9.1. Generate public and private keys	53
9.2. Migration to KSL For Salesforce plug-in version 2.2	55
9.3. FAQ	59

Terms of use of this guide

The rights to use the software described herein are assigned under a license agreement and this guide may only be used or copied in accordance with the terms of the contract.



The information in this guide is subject to change without notice.

The reproduction or transmission of the information in this guide is limited to internal use by the customer and for the sole purpose of proper use of the software. Any other reproduction or transmission is prohibited without the express written permission of NAELAN.

This guide is provided by NAELAN for information on the software delivered. It does not in any way constitute a contractual commitment both on the features indicated and in their implementation.

Unless otherwise stated, the companies, names and data used in our examples are fictitious ; any reconciliation with real companies or entities would be the result of coincidence.

Typographic conventions

<i>Text file sample</i>	Code example or file settings
	Note, information
 Chapter, Guide	Reference to another guide or chapter
<i>Example</i>	Example, variable, code extract
Keyword, label	Keyword, application label, important item

Contact Naelan

Headquarter - 4 rue Claude Chappe
69370 Saint-Didier au Mont d'Or France
+33 4 37 59 81 40
www.naelan.com
support@naelan.com

Office in Paris - 4 Place Louis Armand
75023 Paris France
+33 1 72 76 80 67
contact@naelan.com

1. About KSL for Salesforce®

KSL for Salesforce® is a plug-in that enables the Salesforce® solution to communicate effectively and securely with the KSL servers, to generate, edit, personalize, archive, and/or deliver documents and e-mails.



This plug-in provides several key functions, which are very interesting to secure document and e-mail generation and delivery:

- On-demand creation of personalized documents and e-mails and their delivery, fully integrated with Salesforce® objects
- Automatic production of KSL documents and e-mails, via the Salesforce® process builder
- Documents and e-mail management and archiving
- Document and e-mail template design based on shared components

i In this guide, the term "document" applies to all communications generated by KSL.

i The explanations in the guide often use the example of the Opportunity object, which is the Salesforce® object implemented by default for the plug-in. However, they apply to any standard Salesforce® object, such as Account, Contact, or any Custom Object

1.1. Document and e-mails generation and delivery

a. Template selection by the end-user

The document or e-mail generation starts with the selection of a template in the Document and e-mail templates view.

- This view is a KSL component that appears in Salesforce®
- It is configured for different Salesforce® object (for example, opportunity and campaign)
- It presents the list of available and authorized templates in a tree structure
- These access permissions depend on the user's profile (KSL role)

b. ... or push-button

The targeted document or e-mail template is directly associated with an action button available on the


Salesforce® object.

Authorization to access the targeted model remains dependent on the user's profile.

c. Personalized document generation thanks to Salesforce® data

Each document template is set to be generated on a predefined launch mode and document format.

- Transactional launch mode (on demand): the user directly obtains the final document, whatever its format, PDF most frequently, but also office formats such as DOCX, XLSX and PPTX
- Interactive launch mode: the document is presented to the user in html form, with some customizable content and potentially re-organizing. This launch mode also allows you to keep and version the document in PDF format at different steps of its customization.

 The KSL Interactive Document interface also has a PDF preview; during or at the end of the personalization phase, it allows the user to view and correct, if necessary, the document before its distribution.

d. Personalized e-mail bodies or e-mailings thanks to Salesforce® data

Each e-mail template is characterized

- By the *mail* launch mode
- By a return mode, which is defined by the *MailReturnMode* parameter equal to 1 for sending a single e-mail through Salesforce or 2 for creating an e-mailing

In the case of a single e-mail, the HTML body is personalized by the user and inserted automatically in a Salesforce® e-mail for a manual sending

In the case of an e-mailing, the HTML body of the e-mails is personalized by the user and KSL processes the transmission of the e-mails to an external e-mail send and track service (not provided by Naelan)

1.2. Automatic production of documents and e-mail KSL, via Process Builder

In the Salesforce® KSL Administration application, a dedicated configuration interface allows you to graphically define call classes of models of KSL documents, as well as their distribution method:

- Selecting one or more KSL document templates (regardless of format), associating it with the targeted Salesforce® object
- Selecting PDF annexes
- Selecting a KSL or Salesforce e-mail template or manually complementing the body of the e-mail
- Sending selected documents by e-mail and tracking the shipment.

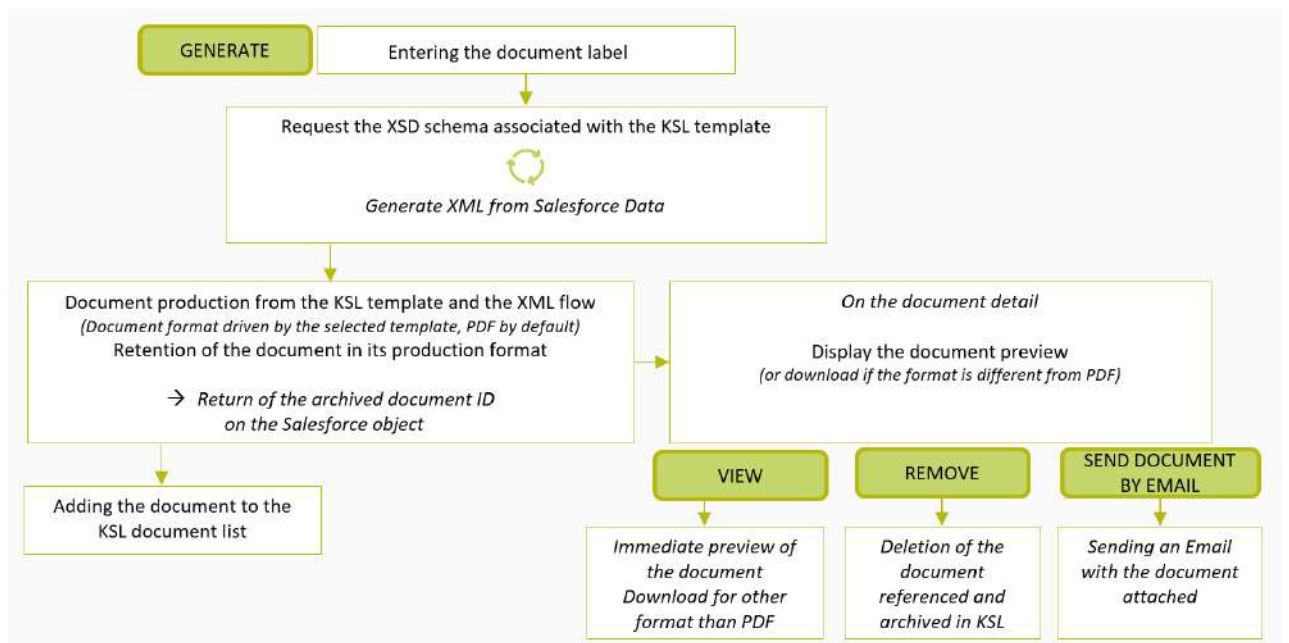
This KSL production and distribution process is directly called by the Salesforce #! {Corps1_Puce}® Process Builder to be associated with a change of state in a workflow or any other automated process. It is a task started asynchronously.

The process requires no user intervention. The latter may, however, observe the sending trace or consult the document or documents produced on the Salesforce® object concerned by the automation.

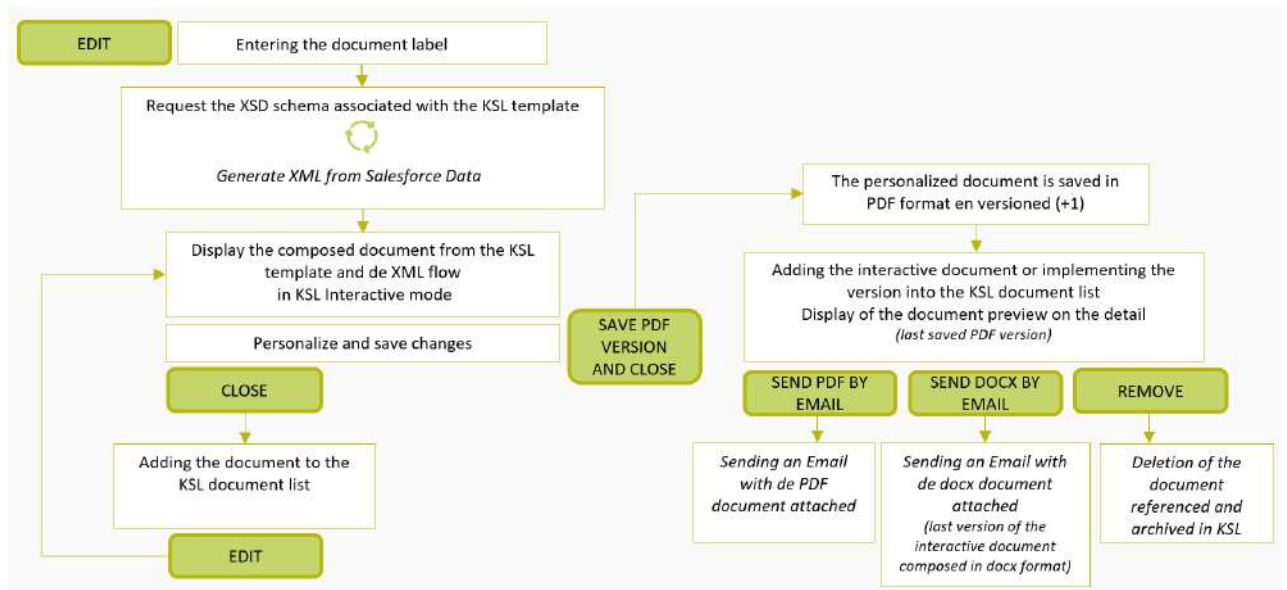
1.3. Management of documents and e-mails created by KSL

a. Personalized documents created by KSL

Document on demand - Process



Interactive document - Process



i Document storage is provided by the KSL solution: document are archived on KSL servers in a SaaS mode

b. Personalized e-mail bodies created by KSL

- The e-mail body of a single e-mail generated after an interactive personalization, is not archived by KSL
- This e-mail body is automatically inserted into a Salesforce® e-mail that is sent and saved in the activities of the object in Salesforce®
- It can be transferred to a new recipient

c. Personalized e-mailings created by KSL

- E-mailings are not accessible from Salesforce® after they are created and launched.
- An task is created in the Salesforce® object for this e-mailing as soon as it is transmitted to the KSL server
- The e-mailing is sent by the KSL server to an external e-mail send and track service asynchronously

d. Sending by e-mail a personalized PDF document created by KSL

- A personalized PDF document PDF can be send by e-mail
- In this case, a Salesforce® activity is created by KSL containing the message sent: the sender, the recipients, the subject and the message body written by the user and its sending time
- This activity is associated to the object used to create the document
- The attachments and especially the KSL PDF document are accessible in the Attachments view
- The documents in attachments are stored in Salesforce® and not in KSL: when sending a KSL document

by e-mail, the document is archived in KSL when it is created and stored in parallel as an attachment of the e-mail in Salesforce®

e. Salesforce® data allowing to personalize documents and e-mails

- These data come from different Salesforce® objects
- They are transmitted in XML format automatically and seamlessly
- Document and e-mail templates are based on an XSD schema corresponding to the XML data
- XSD schema and XML sample can be created by the Salesforce® administrator thanks to the KSL Administration application

1.4. Document and e-mail template design based on shared components

a. A centralized repository containing resources shared by templates

- This repository is centralized and collaborative
- It contains all the components necessary to document and e-mail templates: text zones, groups of text zones, images, PDF appendices and style sheets
- It contains all the document and e-mail templates
- Each component (resource) is shared by the templates
- The repository management can be done from Salesforce® thanks to KSL Administration application or thanks to KSL Studio
- Each text content can be multi-brands, multilingual and / or multi-institution

b. Component and E-mail template design via KSL Administration

- For designing components, please refer to the KSL Office user guide
- For designing e-mail templates, please refer to the KSL Email designer user guide
- For designing document templates, refer to the information below

c. Document template design thanks to KSL Studio tool

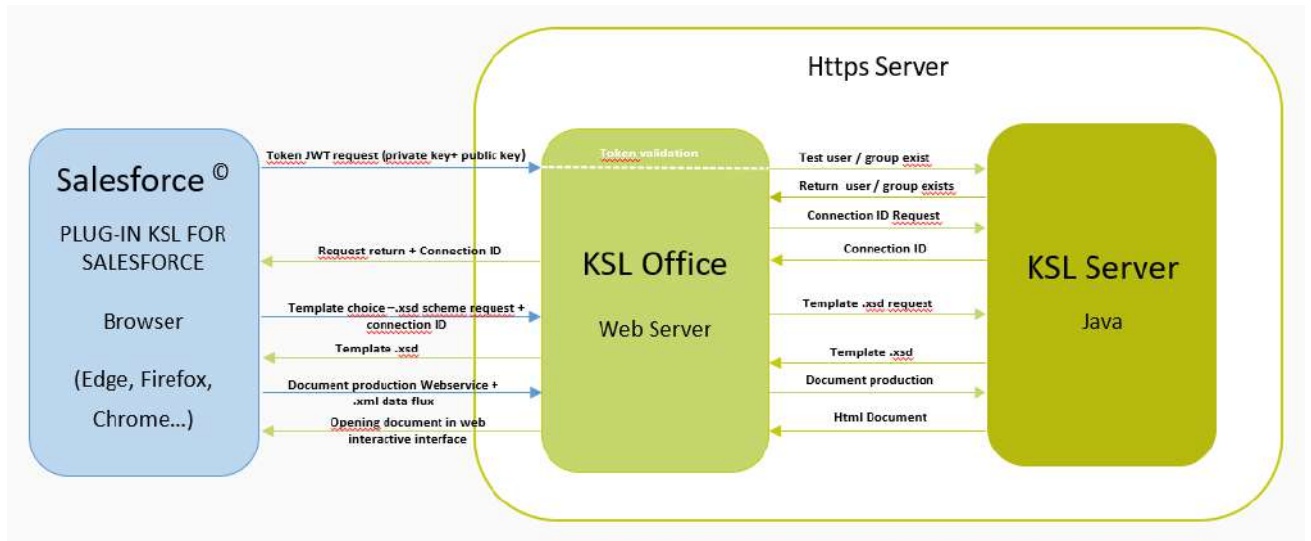
- KSL Studio is a Windows specific design tool you have to install for creating your templates
- This tool also allows to manage the repository components
- This is a collaborative tool for creating simple or complex components, rules and document templates from an XSD schema in connection with the KSL centralized repository

1.5. Technical flow diagram

The diagram below shows the technical components of the KSL for Salesforce plug-in and the details of their

dialog, from the connection to the document generation, illustrated by an example.

Illustration of the launch of an interactive KSL document



2. Installing KSL Suite from Salesforce® AppExchange®

2.1. Technical prerequisites when installing the KSL plug-in

To install and use KSL for Salesforce®, one of the following Salesforce edition is required:

- Enterprise Edition (EE)
- Unlimited Edition (UE)
- Developer Edition(DE)
- Performance Edition
- Professional Edition (with API access option)

Note that the Essential edition (EE) is not supported, because this version does not have API access.

Other prerequisites:

Salesforce® Products	<ul style="list-style-type: none"> ■ Sales Cloud ■ Service Cloud
KSL Products	<ul style="list-style-type: none"> ■ KSL for Salesforce (plug-in) ■ KSL Server
KSL Environment	<ul style="list-style-type: none"> ■ Remote site URL ■ Office app access URL ■ KSL project name ■ Public Key (oAuth2 connection) ■ Private Key (oAuth2 connection)

Compatibility between KSL Server and KSL for Salesforce plug-in versions:

KSL for Salesforce plug-in version	KSL Server version
1.5	8.0.1.1
1.6, 1.7 and 1.8	8.0.2.3
2.0, 2.1, 2.2 and 2.3	8.1.1.5
3.0	8.1.2.7
3.1	8.1.3.5

2.2. Installing KSL from Salesforce® AppExchange®

KSL plug-in is available in AppExchange®.

To install the KSL for Salesforce® plug-in:

1. Search the KSL for Salesforce plug-in in AppExchange®

2. Click on Get It Now and follow the instructions to install KSL Suite for Salesforce®

3. Specify where you want to install the KSL for Salesforce® plug-in:

- Click Install in production to install in a production environment
- Click Install in sandbox to install in a Sandbox

4. Read and accept the terms and conditions:

- Select I have read and agree to the terms and conditions
- Click Confirm and Install

As a best practice, and to avoid access issues for end users, you must install the package for the Admin profiles only, by selecting Install for Admins Only. It enables to use the permission set delivered in the package to grant the rights to the appropriate users.

5. Then click Install:

- Click on the following button to choose the Install for Admins Only option



- Provide System Administrator credentials to install an application from AppExchange®
- After installation, configure the KSL Suite application, including integration settings for third-party applications

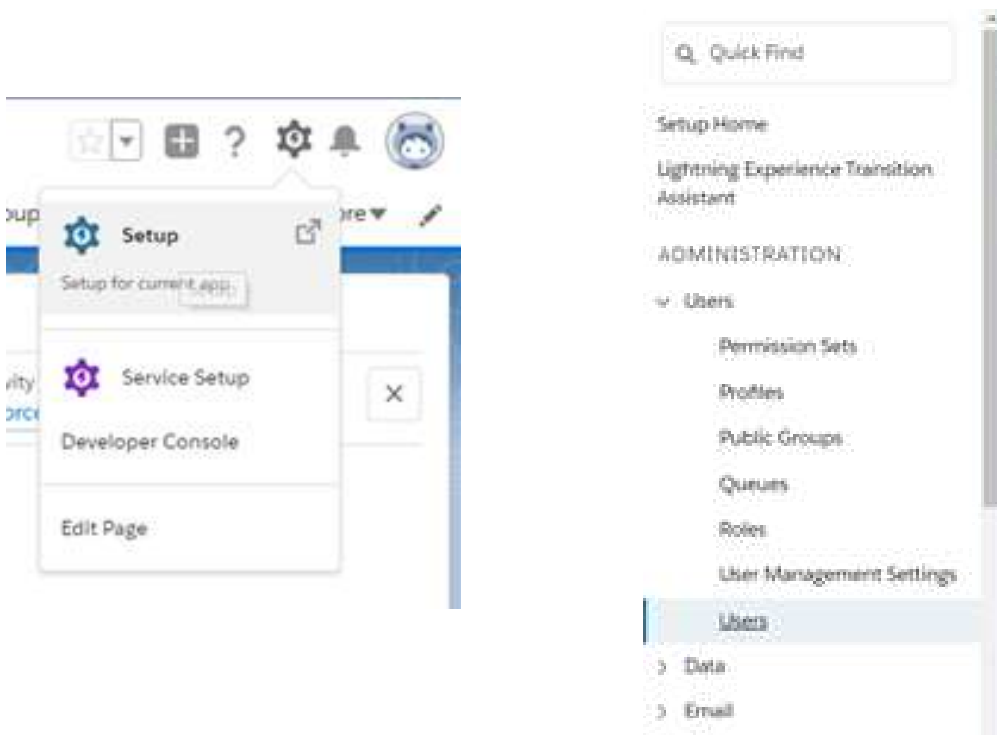
3. Configuring KSL for Salesforce®

3.1. User configuration

No user has to be created in KSL: KSL for Salesforce® relies on the users already created in Salesforce®.

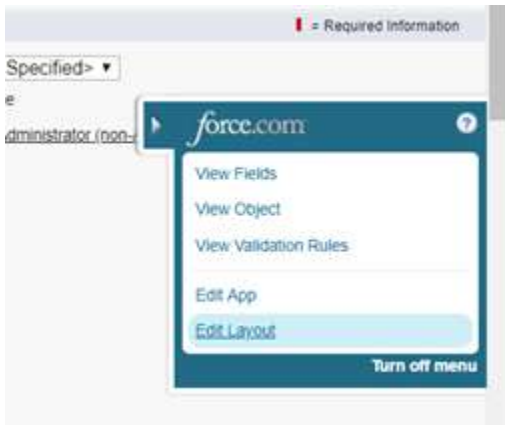
The first step in the configuration of KSL for Salesforce is to configure existing users by granting them rights on KSL For Salesforce® app:

- Grant KSL Suite role to give users rights to create, modify, generate or delete documents on KSL server.
- Select Setup to access to the user administration, then select ADMINISTRATION > Users > Users.



- Click Edit to select a user.

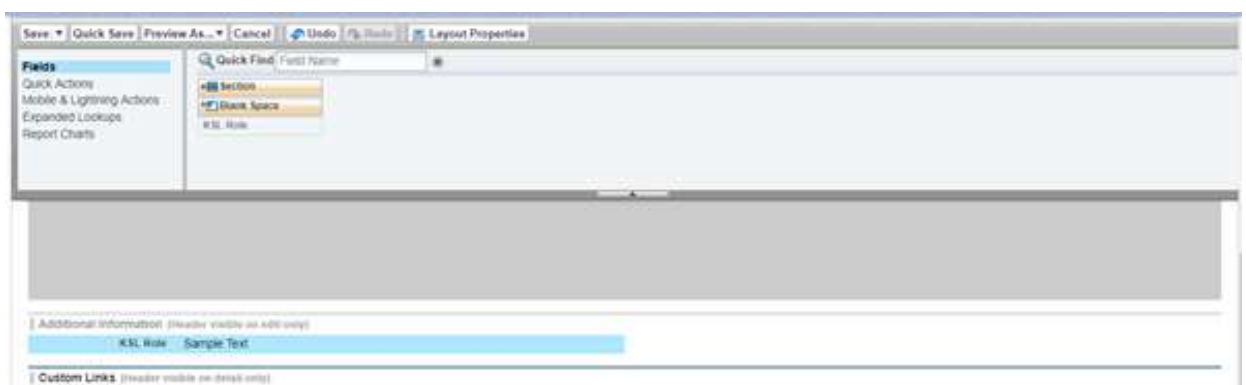
Action	Full Name ↑	Alias	Username
Edit Login	BONNETON_PASCAL	PRONN	salesforce@naelan.com.testolugin
Edit - Record 2 - BONNETON_PASCAL		Chatter	chatty.00d110000000ntaeau.fbm55de2pyna@chatter.salesforce.com
Edit	unused	unused	unused@unused.fr.testolugin
Edit Login	User_Admin	AUser	coa.n30xodvyywhh.ds6imvduhyxo@naelan.com.testolugin
Edit Login	Vérité_Corine	cverite	verite@naelan.com.testolugin



- Click on the arrow at the right of the screen and select Edit Layout. The user layout page is displayed (User Page Layouts).



- Add the field KSL Role (KSL_Role__c) to your User Page Layout.
- Drag and drop the KSL Role available field under Additional information and Save.



- On the user's details, set the KSL Role based on the rights you want to give him; this role corresponds strictly to the group set in KSL Admin (not in KSL Administration).

The screenshot shows the 'Users' setup page in Salesforce. The 'Additional Information' section contains a dropdown menu for 'KSL Role', which is highlighted with a red arrow. Other sections include 'Mailing Address', 'Single Sign On Information', 'Locale Settings', and 'Approver Settings'.

- In Users > Permissions Sets, select KSL User, click Manage assignments, and assign it to each user that will use the KSL for Salesforce® features.
- Select KSL Admin, click Manage assignments and select each user that will use KSL for Salesforce®. These users will have access to the resource repository and will be able to generate XSD/XML data files.



i KSL Admin permissions are an extension of KSL User . A user with this permission level must be a KSL User to access to the KSL plug-in features.

3.2. Configuring the communication between Salesforce® and KSL servers

a. Remote site

KSL for Salesforce® communicates with a KSL server; this server processes coming from Salesforce®

requests and generates documents and e-mails. To allow this communication, you must declare a remote site in Salesforce® with the remote server information.

In Setup, look for Remote Site Settings.

Create a New Remote Site:

- Remote Site Name: name of the remote site
- Namespace prefix: ksl
- Remote Site URL: (example: <https://hostname:XXX> with XXX the https port number)

Remote Site Edit

[Help for this Page](#)

Enter the URL for the remote site. All s-controls, JavaScript OnClick commands in custom buttons, Apex, and AJAX proxy calls can access this Web address from salesforce.com.

b. Connexion parameters

Select the application KSL Administration from the Salesforce® App Launcher menu.

Setup > Auth. Configuration

This tab allows to configure the communication between Salesforce® and KSL server :

KSL Administration Setup Content repository

Auth. Configuration XSD Generator

Information
For security reasons, some informations are not visible.

* KSL Server endpoint
https://naelan.haxro.fr:8443/ksl_office8/

* KSL Project Name
PM_FOR_SALESFORCE

Token - Audience field override
https://test.salesforce.com

* Token - Public Key
3MVG9lcxCTdG2VbxxxxpFguzMWwxYprxTbY2.jTfxxxx_jbykRZξ

* Token - Private Key
Complete this field.

Save

- KSL Server endpoint: URL which allows to communicate with KSL Office application ; the URL must end with the reference of KSL Office application, example: https://host21.naelan.fr:8443/ksl_office/
- KSL Project Name: name of the KSL project which hosts templates and resources. This project will be present on the KSL remote server, (example: KSL_FINANCE)
- Token - Audience field override: this field can be completed, if necessary, with the audience field defined in the KSL parameters, usually an URL (example: https://test.salesforce.com).
- Token - Public Key (clientId): a couple "public/private key" is necessary for oAuth2 authentication. This field contains the same public key than the one configured on the remote KSL web server. This key is used by the KSL server to control that the Salesforce configuration is authorized.
- Token - Private Key (clientSecret): a SHA256 256bytes private key which is used by the KSL remote server to decrypt the identification JWT token. This key is hidden after saving the settings and must be entered at each parameters update.

Additional informations are given on tools that can be used to generate public and private keys in the appendix of this document.

4. Produce on demand a KSL document from a Salesforce® object

KSL allows to produce and personalize documents and e-mails, on demand, from Salesforce objects.

These processes cover the following functions:

- Production of a document in all formats, for example PDF or docx, with immediate return of the document.
- Production of a personalized interactive document, with content to be interactively modified by the user. The finalized document produces a PDF document. In this case, a version management is applied for the PDF document.
- Creation with or without interactive personalization of a responsive e-mail body for immediate delivery.

Whatever the document format, integration is possible in two ways:

- Via an action button for quick access to a pre-targeted document model.
- Via a KSL custom object which displays a part of the library of document templates dedicated to the current Salesforce® object.

The installation of the package provides KSL custom objects directly applicable to the Salesforce® object Opportunity. This setting can be used as an example to add the documentary production objects to any Salesforce® object.

4.1. Associate a KSL Template Library with a Salesforce Object

In Setup > Object manager, click KSL Document object (ksl_Document__c), then click Fields and Relationships.

Just add a lookup relationship field in this object for each Salesforce object on which you want to add a list of KSL document templates.

The Opportunity field is already present because directly installed by the package.

If, for example, users need to access a list of templates from the Account object

- Click New
- Step 1: On the choice of the type of field, tick Lookup Relationship > Next
- Step 2: On the selection of the related object, select Account > Next
- Step 3: Enter the label and the name of the field (KSLForAccount for example) > Next
- Step 4: select the field-level security - profiles list proposed by default > Next
- Step 5: The KSL Document Layout which lists the KSL documents is already ticked in the reference fields for the presentation of the page > Next
- Step 6: Adding personalized associated lists - the Account page layout is already checked; click on Save.

The KSL objects in the package are now associated with the Salesforce object; to finalize the configuration, position the KSL templates component on the object layout.

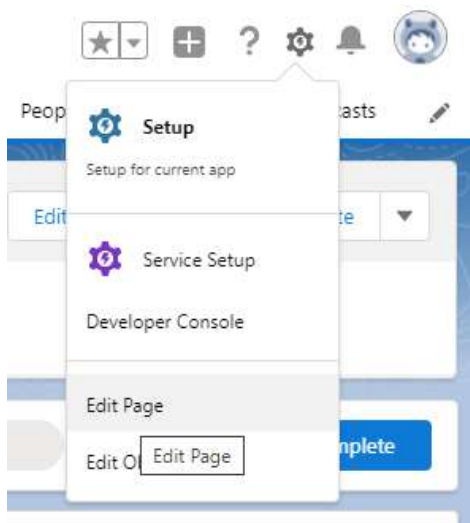
4.2. Integrate a library of KSL models

To generate a KSL Document from a Salesforce® object, configure the object layout to display the KSL For Salesforce® elements.

The component Document templates displays the templates available in the KSL Project configured in the KSL Administration application. The user can then generate a KSL document based on a selected template and its launch mode: e.g. PDF on demand generation or document interactive edition.

The following example gives an example of how to configure the component on the Opportunity object page layout. The process is the same for any standard or custom object in Salesforce.

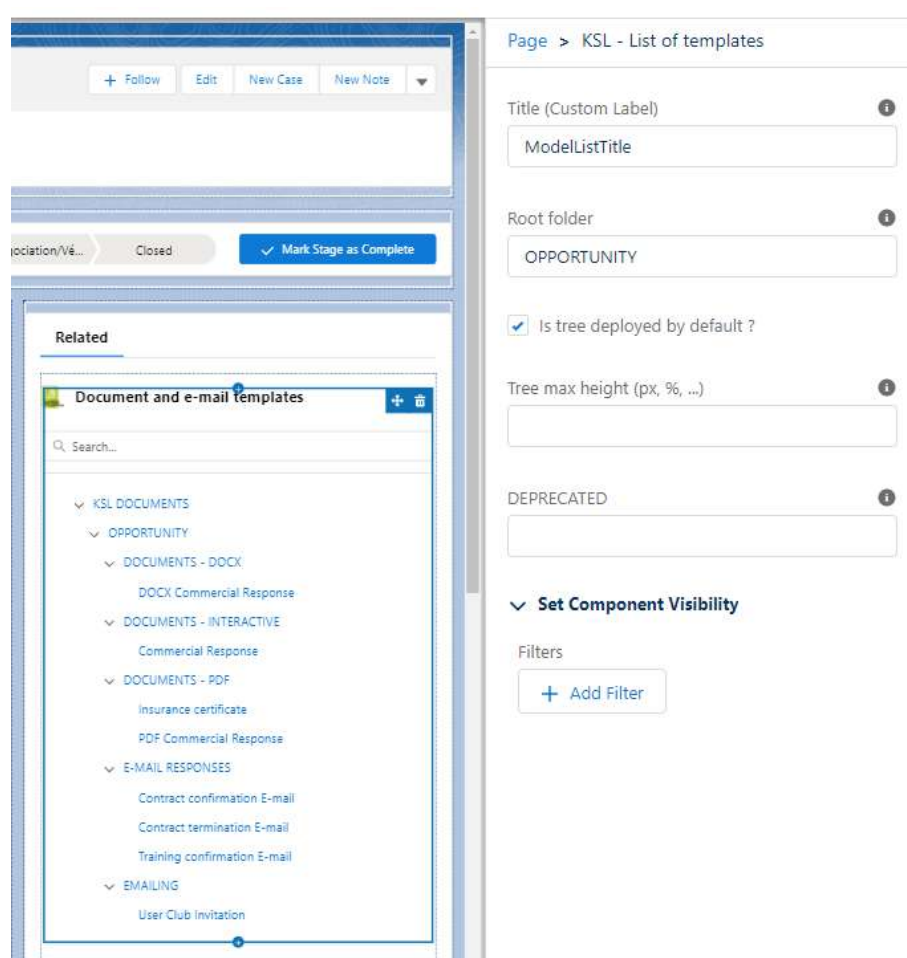
- Go to the detail page of an Opportunity (Opportunities tab), click on the Setup icon and then click on Edit page.



- From the Custom Managed list (on the left), drag and drop the lightning component KSL - List of templates in the page.

The screenshot shows the Salesforce Lightning App Builder interface. The main view displays an 'Activity' section with a list of activities and a 'Related' section showing a classification tree of document and e-mail templates. The right-hand panel shows the configuration options for the component, including Title, Root folder, Is tree deployed by default?, and Tree max height.

- In the option layout of the component (on the right), configure the following options:
 - Title: name of the component record. The entered value must be a Custom Label. The Custom Label ModelListTitle is the object title given to the component with its translations.
 - Root folder: It is possible to filter the classification tree of the KSL project where templates and resources are stored, by indicating the root node to display for the object. Your user must, of course, have the necessary permissions on the node and its potential branches. By default, this parameter is set to OPPORTUNITY for the Opportunity object and CAMPAIGN for the Campaign object.
 - Is tree deployed by default? check this box if the classification map must be deployed by default on this object; this choice depends on the number of templates available and their organization.
 - Tree max height (optional): maximum height of the component on the record page (pixel, percentage,...)



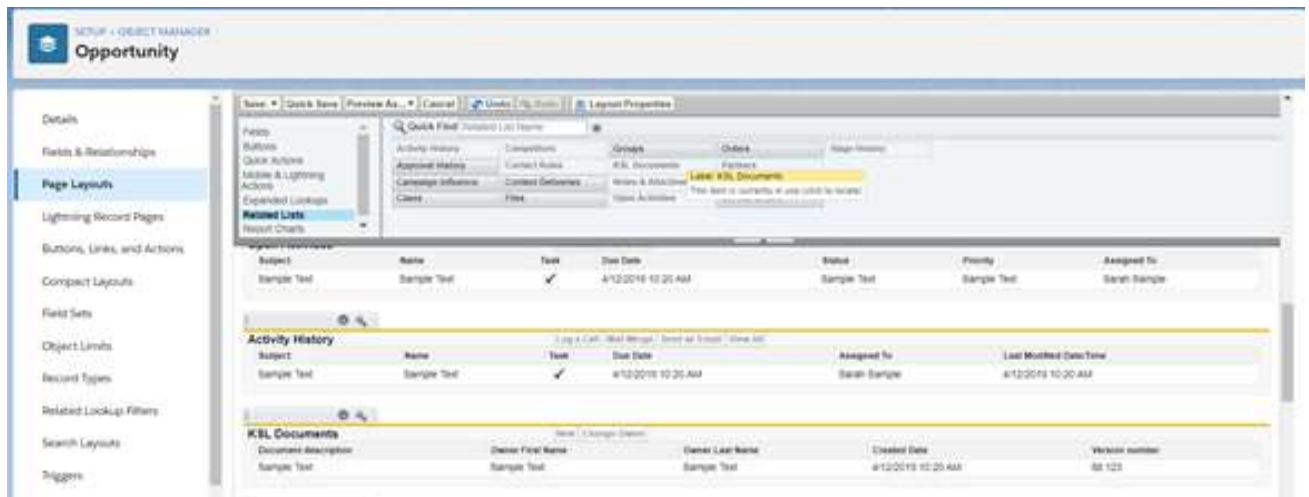
- Set Component Visibility conditions if the document templates window is displayed or hidden. With a filter on a field of the opportunity for example: for example, it is possible to insert the KSL - List of templates component several times on the page, with a different root folder and condition for each component; this allows to provide the right document templates for the current functional context.

i *The notion of visibility allows for example to display a different list of models for two types of cases. To do it, create as many List of templates components as request types and a suitable filter for each: for example, a component for post-sales cases and a component for pre-sales cases.*

Save then activate (Activation) the lightning page before closing the window.

The list of KSL documents linked to the current object is already available on the page, under Related tab. To add the list of the available KSL documents of the opportunity,

- From configuration > Setup
- Tab Object Manager > Opportunity > Page Layouts
- Click Opportunity Layout

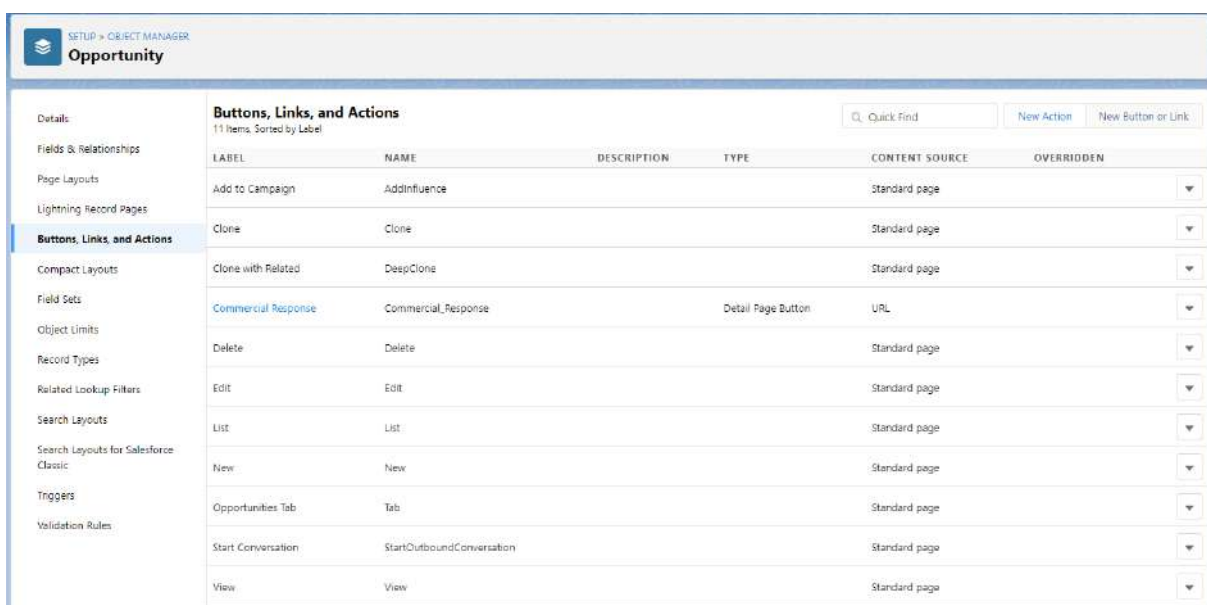


- Select Related Lists
- Select Ksl Documents and drag and drop the list at the 1st position in the existing related lists for the opportunity ; click on Save ; the list appears in the opportunity on the right side in Related tab, under the list of document templates

4.3. Integrate an action button (push-button)

Available and actionable in the Salesforce® object, it is the ideal shortcut to obtain the selected document or e-mail in one click. This setting is adapted to document and e-mail templates that are called frequently.

a. In the setup, create a new button



Object Manager > *my object* > Buttons, Links and Actions > New Button or Link.

Edit Opportunity Custom Button or Link
Commercial Response Help for this Page

Custom Button or Link Edit Save Quick Save Preview Cancel

Label:

Name:

Description:

Display Type: Detail Page Link [View example](#)
 Detail Page Button [View example](#)
 List Button [View example](#)

Behavior: [View Behavior Options](#)

Content Source:

Quick Tips: [Getting Started](#), [Saving Buttons & Links](#), [Operators & Functions](#)

Select Field Type: Insert Field: Insert Operator:

Functions:
 ABS
 ADDMONTHS
 AND
 BEGINS
 BLANKVALUE
 CASE

`/lightning/cmp/ksl__ModelProcessorAura?c__recordId={!Opportunity.Id}&c__serviceName=BP_SolutionAndServices_Portrait&c__editMode=generate&c__name=Commercial Offer {!Opportunity.Name}`

Display type: Button to the Detail Page

Behavior: Display in a new window

Content source: URL

In the multiline field, use the following formula:

```
/lightning/cmp/ksl__ModelProcessorAura?c__recordId={!Opportunity.Id}&c__serviceName=BP_SolutionAndServices_Portrait&c__editMode=generate&c__name=Commercial Offer {!Opportunity.Name}
```

And update the elements:

- `c__recordId`: identifier of the target Salesforce® object, helping you with insert fields utility.
- `c__serviceName`: name of the KSL document or e-mail template (= name of the KSL service),
- `c__editMode`: launch mode: generate for a PDF on demand, edit to get an interactive document, Email to get a responsive e-mail. This mode must be authorized for the selected KSL service
- `c__name`: name of the generated document (spaces and accents allowed); it can be changed from the list of fields.
- `c__emailMode`: not mandatory. This parameter is only associated with email mode. Select 1 to specify that a single personalized e-mail (e-mail body) will be created and send by Salesforce, or 2 for an e-mail sent by KSL server.

i The documentary or e-mail production service called (`c__modelName`) must be consistent with its production mode (`c__mode`), and the advanced production mode for e-mails (`c__emailMode`).

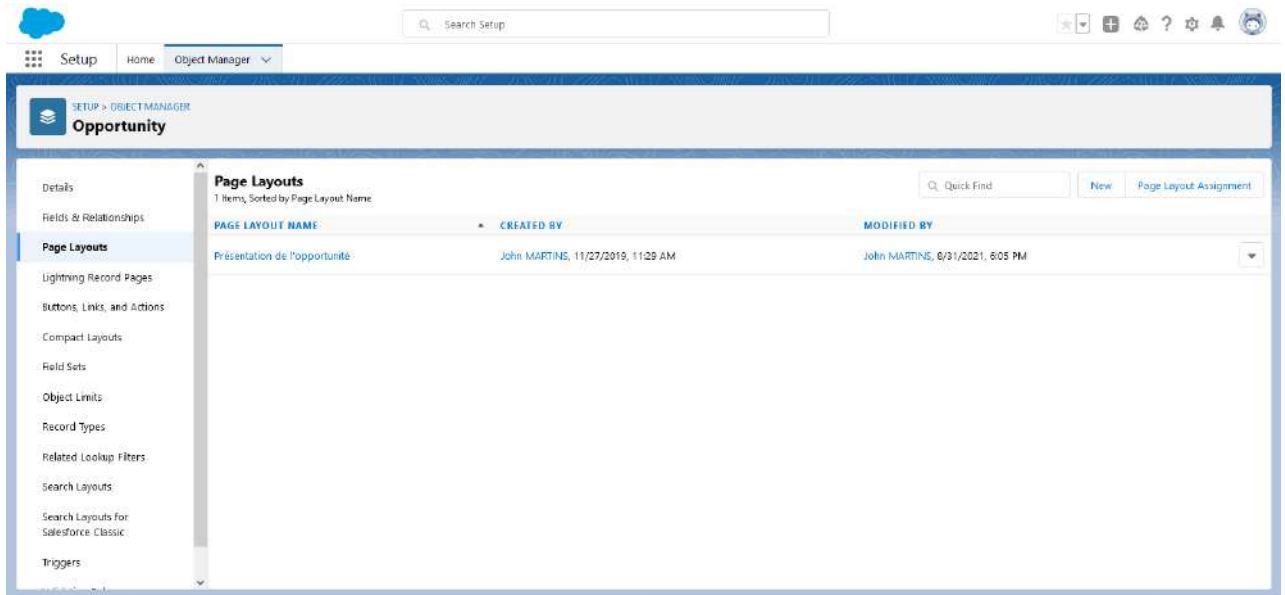
To ensure proper process, this setting and the one defined in the KSL project must be identical.

An e-mailing service (multiple e-mails) cannot, in any case, be associated with the emailMode 1 (single

e-mail) for example.

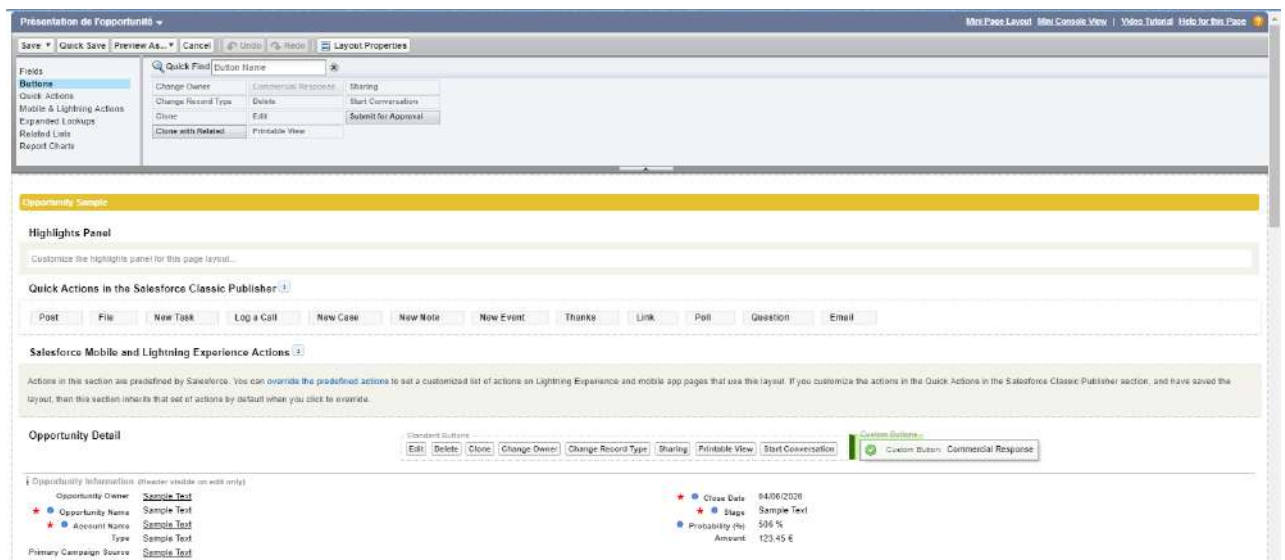
b. Integrate the button into the Salesforce® object

Object Manager > *my object* > Page Layout



Add the button to the selected page layout.

Drag & drop the new button on the object Detail, in the Custom Buttons part, after Standard Buttons.



The new button appears in the quick actions list, on the Salesforce® object.

To add this button in the buttons directly displayed on the object page, personalize the list of Salesforce mobile and Lightning Experience Actions by moving the button to the desired position, at the top of the

list.

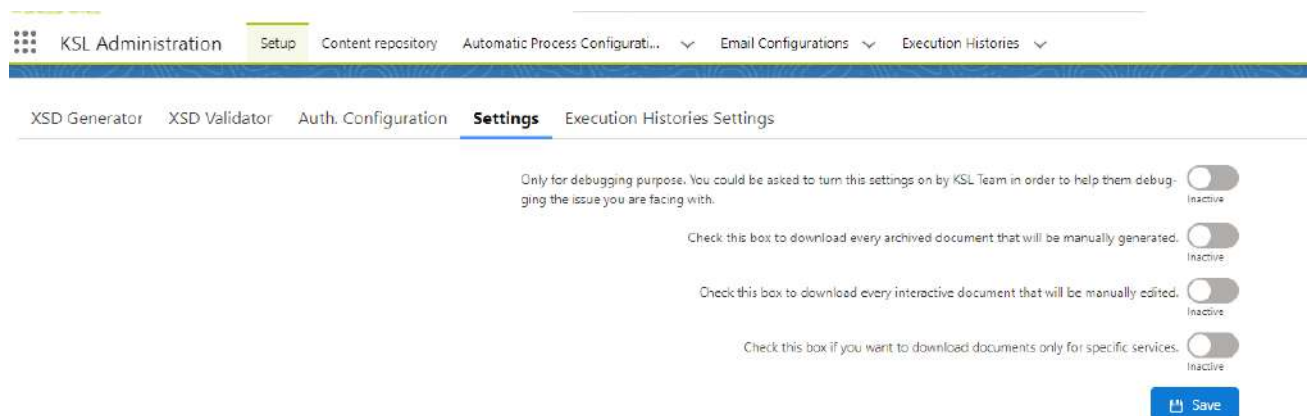
For example, a button *Commercial proposition* can be set in the third position of the visible buttons of the object interface, which allows it to be immediately visible on the object, without open the list.

4.4. Other parameters: retention of transactional documents

KSL Administration > Setup > Settings

Except the first parameter which allows detailed traceability of errors on on-demand documents, this set of parameters is mainly used to force the storage of KSL documents in Salesforce files.

These parameters activate filters (based for example on document template names) which allow to select the documents which will be stored in Salesforce files. The objective of this copy of documents in Salesforce is to make them accessible by external applications (for example an application that allows to send the document in a digital signature process).



- Option 1 / debugging: it can be activated by an administrator in case of a on demand document generation error. This option allows to get more details that the generic error message sent to the user. This option stores the detailed error in the execution traces to facilitate its correction.
- Option 2 / download every transactional documents: this option allows to store all on demand documents (non-interactive) in Salesforce, regardless of the format.
- Option 3 / download every interactive documents: this option allows to store all the PDF documents associated with interactive documents.
- Option 4 / download only the documents generated by referenced templates: this option allows to store in Salesforce (Files), the PDF documents generated by some specific types of documents (for example a contract for a further document signature); in this case, select one or more document templates from the template tree.

Check this box if you want to download documents only for specific services. Active

Services names that are allowed for download. ⓘ

Attestations_RC,
BP_SolutionAndServices_Portrait

Search...

- > INTEGRATION SERVICES
- ▼ KSL DOCUMENTS
 - > ACCOUNT
 - ▼ OPPORTUNITY
 - > DOCUMENTS - DOCX
 - > DOCUMENTS - INTERACTIVE
 - ▼ DOCUMENTS - PDF
 - Insurance certificate
 - PDF Commercial Response
 - > QUOTE PROPOSAL DOCUMENTS

5. Produce a KSL document from a Salesforce® automated process

A KSL process configuration allows to create a automated process that generates a document and if necessary, sends it by e-mail. This configuration can be handled in a the Salesforce® Process Builder or Flow application.

i A Salesforce process, and consequently a KSL process, can be triggered from many different events:

- on a record change, that is, when a new Salesforce object is created or updated,
- on an event, i.e. when the object receives an event message,
- on an invocation when something other than the process invokes it,
- a Einstein Automate process,
- etc.

A KSL process configuration is implemented in the KSL Administration application available in Salesforce®. KSL Administration provides the administrator a step-by-step set-up; the objective is to create a KSL process (a class) that can be directly called on a Salesforce action.

5.1. Configure a KSL process

a. Activate the functionality

In KSL Administration application, select Setup > Execution Histories Settings.

XSD Generator XSD Validator Auth. Configuration Settings **Execution Histories Settings**

Is Automatic Process feature activated ? Includes execution of configurations and cleaning of histories. Active

* How many days before cleaning histories ⓘ
30

Services List : root folder ⓘ
KSL DOCUMENTS

Save

Position the button Is Automatic Process feature activated on Active:

- to activate the KSL process automation,
- as well as to delete the trace history.

Specify the number of days to keep execution traces with the How many days before cleaning A.P.H. field.

Specify the root of the classification tree under which the templates associated with the attachments are available (for automatic email sending). By default, the configuration screen displays all the available templates, excluding the KSL e-mail body templates.

Select Save.

b. Configure a KSL process

In KSL Administration application, select Automatic Process Configurations tab.

The tab lists the KSL process configurations already created, to allow a quick access and modification of these processes.

This tab also allows to create a new KSL process configuration and to configure it step by step. A new KSL configuration consists in a chain of actions, from the generation of the document to its distribution by e-mail as an attachment.

- New allows to create a new KSL process configuration.
- Import allows to import new configurations or modify existing configurations from a .csv file (see the Data Inspector tool available in Chrome on the right arrow, for export).

To edit an existing KSL process configuration, click on the Name of the configuration to access to its details:

- The Details tab gives the main identifiers of the configuration: Name, Related Object Name and KSL API name.
- The Related tab presents the lists of KSL Automatic Process Histories which are associated with the KSL process configuration.
- The Activity view contains the different changes operated on the KSL configuration. All the fields are logged, except the Owner. This view also allows the user to send e-mails or save to-do items.

Click Modify to access to the different settings of the configuration.

Create a new process configuration

Step 1: Basic Information

This step allows to define the generic information of the KSL configuration.

- Enter the Name of the process: only alphanumeric characters can be used; spaces are not allowed; “_” is authorized.
- Select the name of the KSL object linked to the Salesforce® object in the KSL object API name menu list.
- Select the object name which will trigger the processes of the KSL configuration (e.g. the generation of a document), in the Related Object Name menu list.

The Next button allows to validate this step and go to the next one.

Step 2: Document Generation

This step allows to define the necessary settings for the KSL configuration to generate one or more documents. These documents will potentially be attached to the e-mail configured on the next page.

Enter a name for the document that will be generated; this name can be made of text and merge fields. The possible fields are proposed in the Merge Fields area at the right of the window: to add a merge field, select the object then a field in the list and copy/paste the variable name in the Document Name field.

For example, click on the Opportunity, then select its Name field: the {{{Opportunity.Name}}} field name is displayed at the bottom of the Merge Fields area and can be copied/pasted.

The screenshot shows a 'Merge Fields' window. On the left, a list of objects is shown: Recipient, Sender, Organization, and Opportunity. 'Opportunity' is selected and highlighted in blue. On the right, under the heading 'Select Merge Field', there is a search bar with the text 'Search Opportunity merge fields...'. Below the search bar is a list of fields with radio buttons: Fiscal Year, Forecast Category, Forecast Category, Has Line Item, Last Activity, Last Modified By ID, Last Modified Date, Lead Source, Name (which is selected with a blue dot), Next Step, Opportunity ID, and Opportunity Type. At the bottom of the window, a text field contains the merge field code: {{{Opportunity.Name}}}

In the classification tree under the Template Name field, select the KSL template which will be used to produce the document.

Select a new model from the classification map to generate a new document on the same procedure.

One line is created for each selected document template. It is possible to delete an unwanted document by clicking on the trash icon.

Activate the Download Document button so that the document is available in the list of files associated to the KSL document object: with this option, a document generated automatically by the KSL process for a client will be available for the Salesforce user in the file list of the object and also from the Files library.



i Notes:

- *This Download Document option must be active when the document generation is followed by an e-mail delivery; in this context, the Send Document By Email button of the next step is also active, and the document is transmitted as an e-mail attachment.*
- *If the Download Document button is active and the Send Document By Email is not active, the document will be available in the list for a further sending. And a user will be able to send this document as attachment through a Salesforce e-mail with other documents - all the documents available in Files.*
- *If the Download Document button is not active, the document will not be available in the list of files, but it will always be present on the object from which it has been generated via a link to KSL servers.*

The Next button allows to validate this step and go to the next one.

Step 3: Document Sending

This step allows to define the settings necessary to send a generated a document by e-mail.

Progress bar: ✓ ✓ Email Sending Error Management

Send Email (with files if applicable) Active

Email Sending

From: User sending the email

Email Template: Contract confirmation E-mail (KSL - /KSL DOCUMENTS/OPPORTUNITY/E-MAIL RES)

To: {{{Opportunity.ContactId}}}

Subject: Our new proposal - {{{Opportunity.Name}}}

Cc:

Bcc:

Reply To:

Cancel Previous Next

Activate the Send Document by E-mail button so that the KSL process configuration sends the e-mail.

Select the recipient addresses required in the To, Cc and Bcc fields.

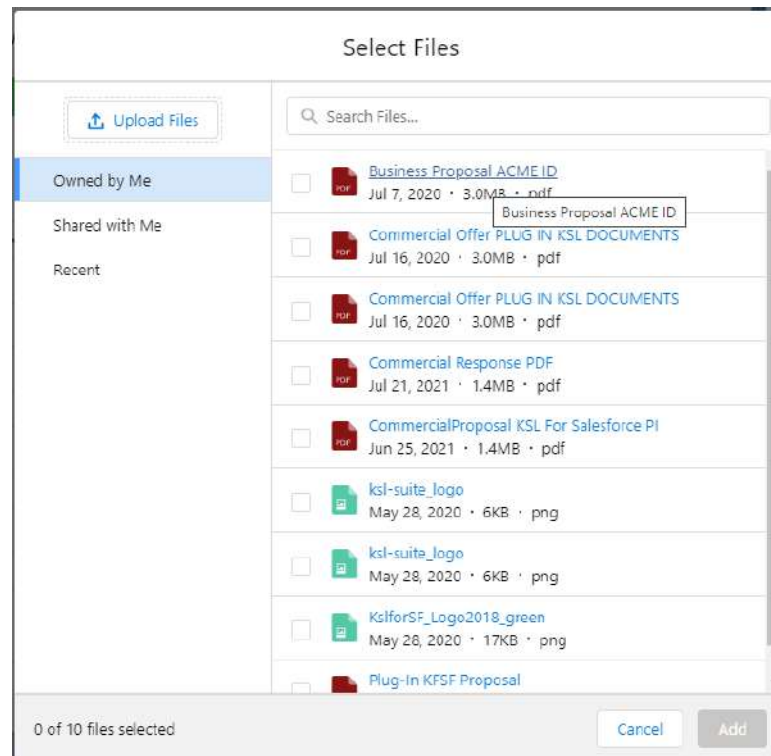
These addresses can be personalized using some addresses of the Salesforce directory, some merge fields or e-mail addresses.

- Enter an email address,
- or reference an already existing contact (in this case his thumbnail is displayed),
- or reference the owner of the object (e.g. the opportunity owner) or a client contact using merge fields.

Attach file

Cancel Previous

It is possible to attach one or more external document (file attachments) to the -mail. These documents can be selected from those available in the Salesforce Files or download from the local user's file system.



Select an Email Template from:

- the list of standard Salesforce® emails,
- the personalized emails created with KSL (in the same list)



The name of the personalized e-mail template can also be specified by the concatenation of text and Salesforce merge fields.

Select the Subject of the email. Like the name of the template, this field can also be changed and personalized with text and merge fields.

The Next button allows to validate this step and go to the next one.



A KSL process configuration can generate a document, link it to an object, and yet sent an e-mail without any document attachment.

Step 4: Error Management

This step allows to define the settings for managing process errors.

Create Automatic Process Configuration

Error Management

Error Management

List of additional recipients ⓘ

In the List of additional recipients field, select some contact addresses (e-mail or contact) to which error messages will be sent, in addition to the object owner address. Usually, this field is used to send error messages to administrators.

Click Save to validate the complete configuration of the KSL process, which can now be used for example by Process Builder or Flow Builder.

5.2. Execute the KSL process from the Salesforce® Process Builder

Access to the Setup of the Salesforce organization then select: Process Automation > Process Builder

The objective is to trigger an email sending when the amount of the opportunity is modified, with the updated commercial offer in attachment.

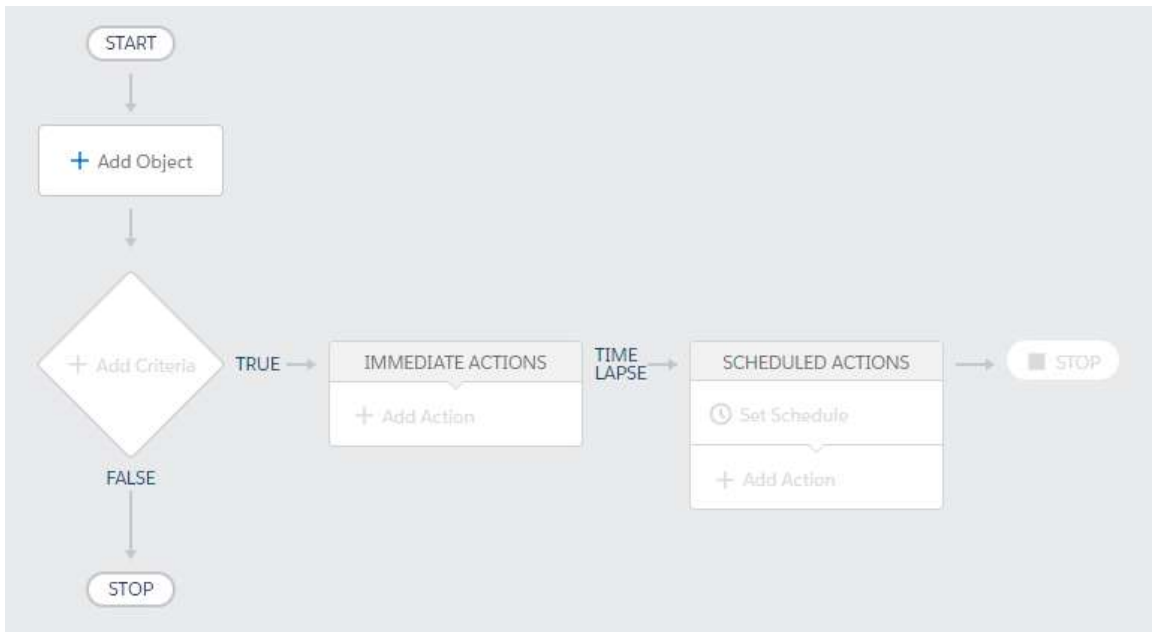
New Process

Process Name * API Name * ⓘ

Description

The process starts when *

- Enter the name of the process -> the name of the API is automatic.
- Enter the description.
- The process starts when... A record changes

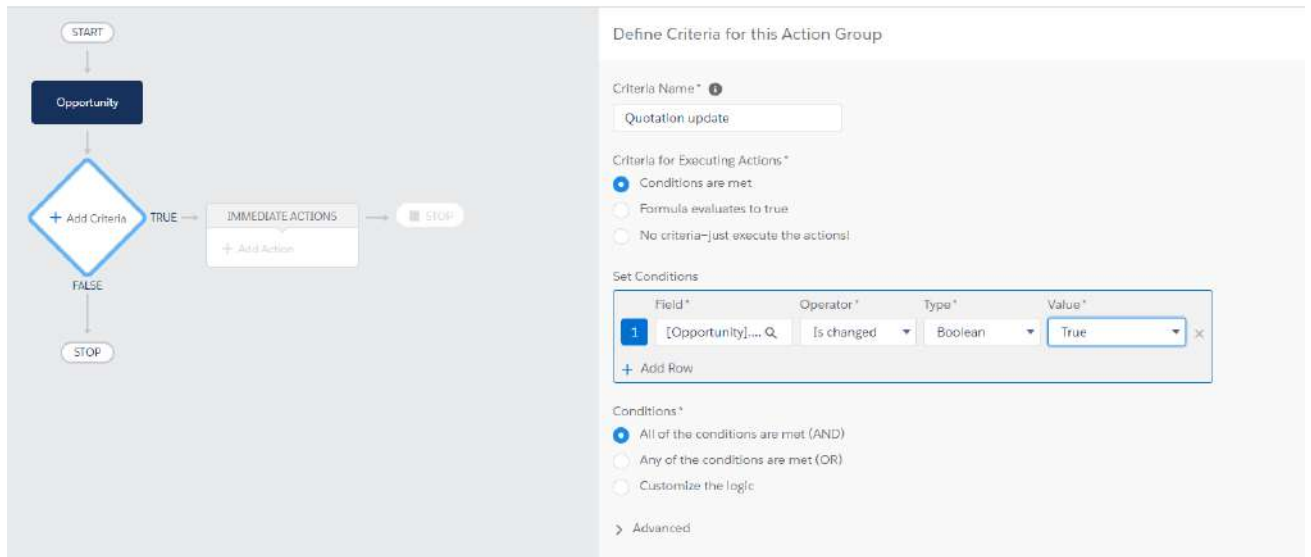


Add Object

- Select the object.
- Indicate if the action is triggered on creation or on creation and update.

i If the opportunity already includes an active automation, it will be reported on this interface.

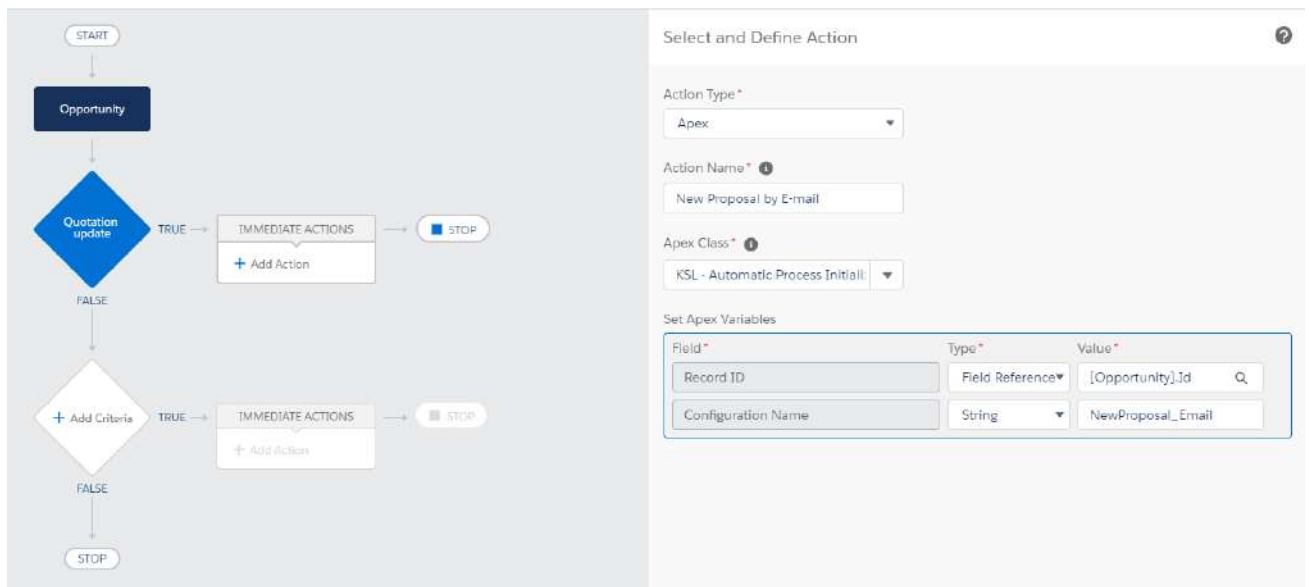
Add Criteria



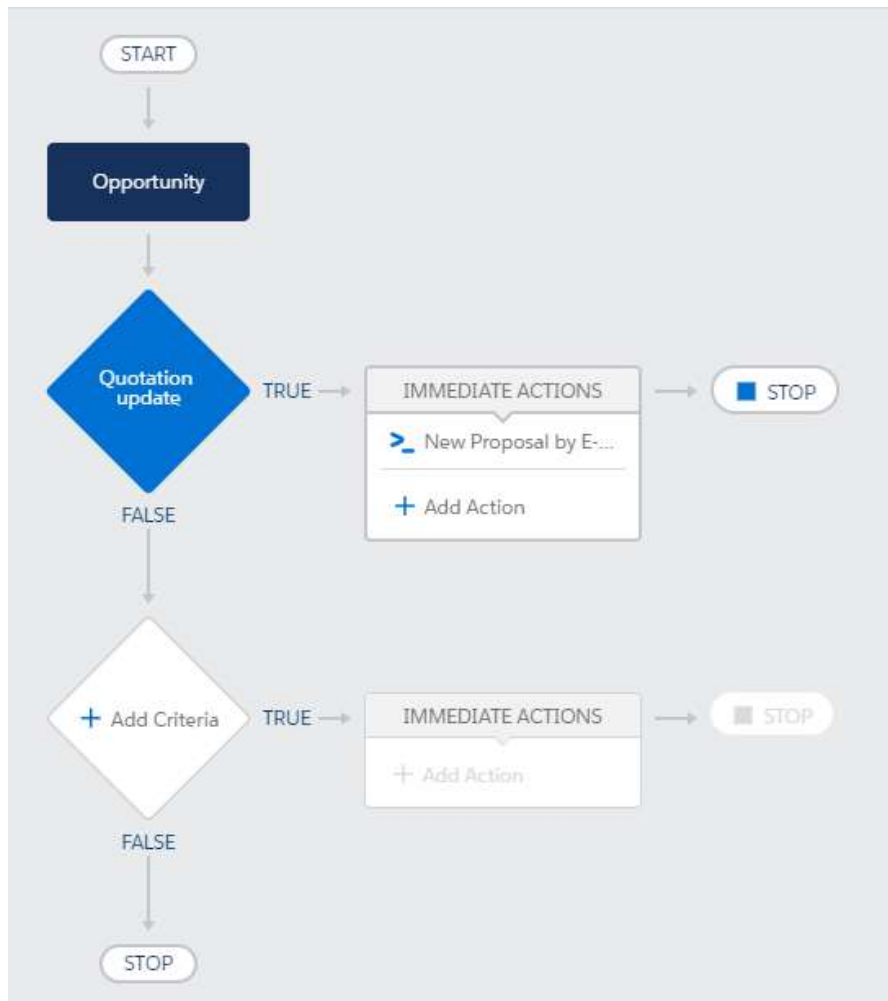
Enter:

- the name of the criterion,
- performance criteria,
- the performance condition(s)

Add Action



- Action type : Apex.
 - Apex Class : KSL – Automatic Process Initializer
- Set Apex variables :
- Record ID : identifier of the current object. Use the help provided in the field (ex : [Opportunity.Id])
 - Configuration Name : KSL process name



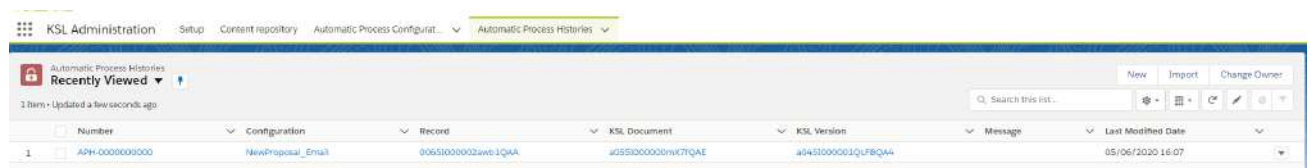
Activate the process via the blue "Activation" button.

***i** When the process has been activated, it cannot be changed; it is mandatory to create a new non-active version of this process for modification.*

The Salesforce® process can be deleted from the list of processes, by developing the process, then delete on the targeted version.


5.3. Control the execution log

In the KSL Administration application, select Automatic Process Histories.



- The last histories appear at first
- The message column is populated in case of error

This list includes:

- the trace of KSL processes launched automatically, in error or not,
- potentially the trace of documents produced by KSL from Salesforce objects when in error, at the condition that this log is activated in the main parameters -  See the next chapter

Each information is a link that refers to the details of the object: process, configuration, Salesforce object, KSL document and Version of the KSL document.

In Sales > Reports > All Reports

- Click KSL.
- Click on All Failed Automatic Process Histories.

Report: Automatic Process Histories All Failed Automatic Process Histories							
Total Records							
1							
	Automatic Process History Number	Configuration	Record	KSL Document	KSL Version	Message	Cause
1	A6H-0000000001	NewProposal_Email	0065000002aueb1QAA			An error occurred during the document generation	No such column: 'TradeStyle' on entity 'Account'. If you are attempting to use a custom field, be sure to append the '_c' after the custom field name. Please reference your WSDL or the describe call for the appropriate name.

6. Pre-configure the fields of the e-mail sending window

6.1. Objective

From a Salesforce object, the user has the option:

- to send a responsive KSL e-mail whose content has been personalized,
- to send a document generated by KSL as an e-mail attachment.

For each of these options, an e-mail sending window opens allowing to personalize the delivery options.

E-mail

*De	<input type="text" value="John MARTINS <pbosalesforce@naelan.com>"/>
A	<input type="text"/>
Cc	<input type="text"/>
Cci	<input type="text"/>
Objet	<input type="text" value="AAA"/>

If this email cannot be displayed, please consult the web

Dear Mrs. Dewailly,

I received your wish to stop the contract and I can confirm your request. Your existing contract will finish at its end.

We hope you enjoyed our services.

The objective of the e-mail configuration is to preset the different parameters of the e-mail window with data that can be personalized by Salesforce fields, in order for users to save time and secure the input.

6.2. E-mail configuration

From the applications list, select KSL Administration > Email Configurations

Click New to create a configuration.

An e-mail configuration can be defined for each Salesforce object: opportunity, account, contact, case...

Select the object linked to the e-mail sending window: the merge fields selector (on the right) will adapt to the object type.

Specify which fields are read-only; these fields will be presented to the user but cannot be modified. It is thus possible to block the sender's address and the reply address, in particular if there are generic addresses for company services (support@..., commercial@...).

- From: sender's address; by default, the address of the logged-in user (user who sends the e-mail), but it is possible to preset a more generic address if tis address is available in the e-mail addresses defined for the organization. Generic addresses appear in the list.

- To: recipient's address; it is possible to type text or preset the the contact's address available on the current object, if his identifier is available in the merge fields of the linked object.
- Cc and Bcc: addresses for copy and blind copy, to be completed on the same model as the recipient's address, if necessary.
- Reply To: if necessary, especially if it is different from the sending address.
- Subject: the subject of the e-mail can be preset with the elements of the current object, the name of

an opportunity, an account, an invoice number for example. In this case, it will be personalized thanks to the merge fields available on the right part.

Select Save

The new configuration appears in the list, identified by the linked Salesforce object.

7. Validation workflow example

In Salesforce®, you can configure validation workflow to validate some records and especially KSL documents.

In this section, you will find a simple example of a validation workflow that you can configure for the KSL (PDF) document. For more information on validation workflow, you can read the Salesforce® documentation.

7.1. Creation of a queue

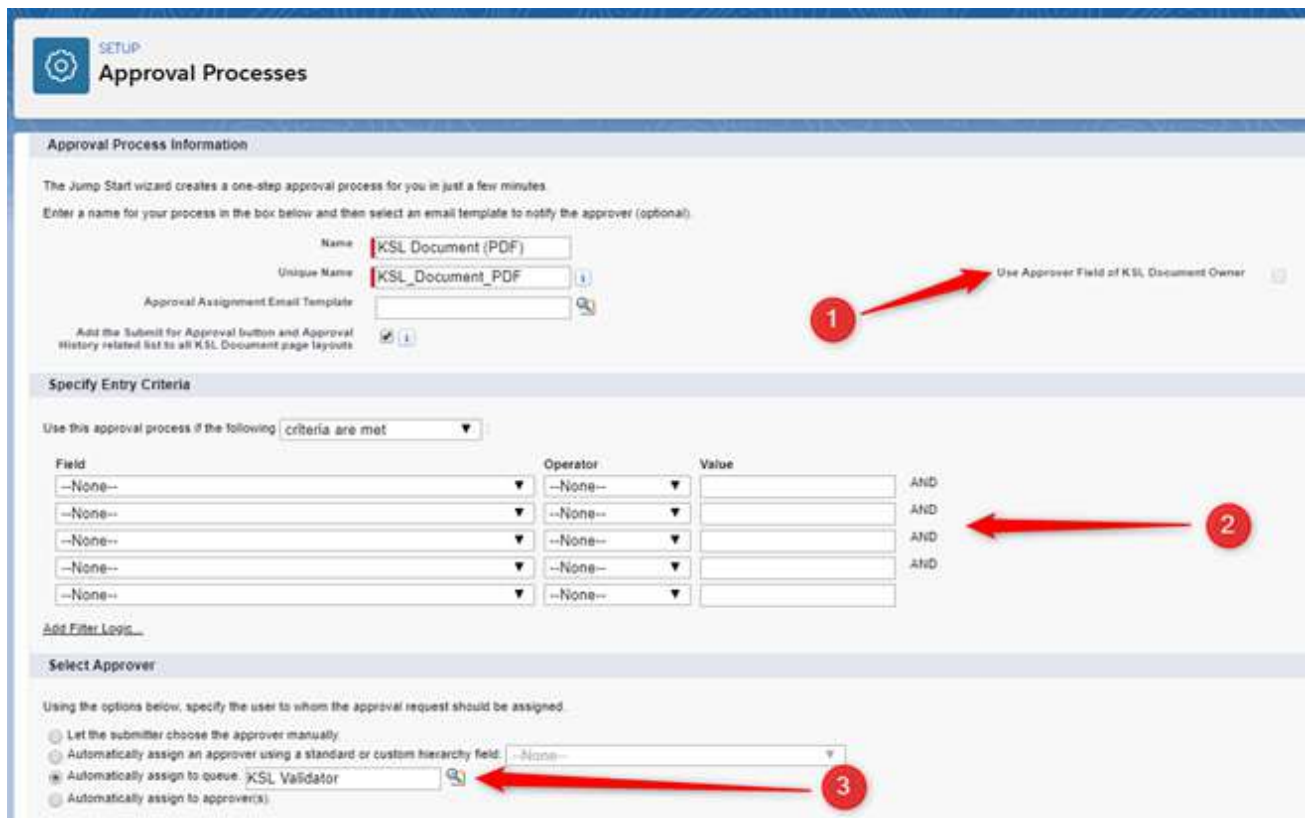
If you have multiple users assigned to validate the document, you can create a queue to manage it. In Setup, find Queues and click the button New.

The screenshot shows the 'Edit Queue' interface in Salesforce. The queue is named 'KSL Validator'. The 'Queue Name and Email Address' section includes fields for Label, Queue Name, and Queue Email. A red arrow labeled '1' points to the Queue Name field. The 'Supported Objects' section shows a list of available objects and a list of selected objects. A red arrow labeled '2' points to 'KSL Document' in the Selected Objects list.

- Choose the name of your queue
- Choose the objects that will be assigned to this queue. In our example KSL Document.
- Choose the members of the queue. These can be users, roles, or public groups.

7.2. The validation workflow

In Setup, search for Approval Processes. Choose the object on which you want to create your validation workflow, in this example KSL Document (PDF), and then Create New Approval Process.



- Choose the name of your validation workflow and the e-mail template to send to the approver if necessary.
- Specify whether KSL documents (PDF) are automatically transmitted to the validation workflow and the condition. Otherwise, the user may manually submit his KSL document (PDF) to the validation workflow.
- Select the approval of the KSL Document (PDF). In this example, the record is assigned to the previously created queue.

On the Approval Process (Layout) screen, you can add new approval steps and choose whether the Salesforce record (the document in the context) should be locked when it is approved or rejected.

8. Configuration of external document storage

This document describes the development and configuration steps necessary to allow the storage of documents generated by KSL for Salesforce in an external document management system (DMS), as an alternative of the usual KSL Suite archiving system.

Archiving documents in an external DMS requires to have a DMS APIs allowing Salesforce to store documents. These APIs are called by a specific APEX class that the company has to develop and which is called by the KSL for Salesforce plugin.

This setup requires to create a specific APEX class, to setup this class in the KSL plugin and permissions and configuration of the external access of the Salesforce organization to the external system.

8.1. General operation

In a standard operation of the KSL for Salesforce plugin, a document created by KSL is stored on the KSL server side and linked to the Salesforce object that created it.

The KSL plugin allows documents to be archived in the customer's storage system (external storage). This system can be a DMS or a digital archiving system.

The display of documents in Salesforce remains identical to a standard KSL for Salesforce operation.

The implementation of this new operation are dedicated to companies that have development and integration capabilities, more specifically capabilities to develop a storage function that is specific to them.

This function is called by the KSL plugin; it must be based on an API that allows documents to be loaded into the external storage system and then returns a unique technical document identifier during the operation.

The function is based on a public APEX class to be created by the company to control the sending of the document to the external system.

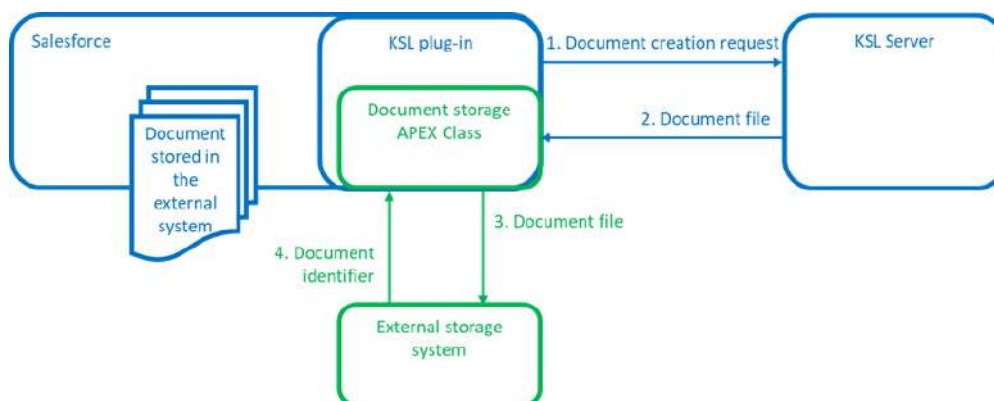




Figure above: principles

1. The document is generated on demand or from a KSL process, or generated from an interactive document.
2. The KSL plugin requests the document (the document is received in PDF format for example).
3. The document and the KSL Document object are passed to this class.
4. The identifier of the document stored in the external system is returned by the class to the KSL plugin. The KSL Document object is created with this identifier. The document is displayed, as in standard storage mode, in the Salesforce KSL Document object.

8.2. Technical description

The KSL plugin calls a customer-developed class that implement the external storage. This class is based on an interface provided by the KSL plugin.

 It should be noted that the class can be called from any context (simple transaction, batch, queue...) with the limitations that it implies. If the KSL plugin generates several documents during the same transaction, the unicity of the class instantiation is not guaranteed.

 It is also important to limit SOQL/SOSL queries and callout number as much as possible. The current operation of KSL plugin does not include DML or any operations preventing the execution of callouts.

Use example of the class:

```
public class SampleArchiveDocumentExtension implements
ksl.ArchiveDocumentExtension.Process {
    public ksl.ArchiveDocumentExtension.Response
process(ksl.ArchiveDocumentExtension.Request request) {
    try{
        /**
         * Do something
         */
        return new
ksl.ArchiveDocumentExtension.Response().success('%DOCUMENTID%');
    } catch(Exception e){
        return new
ksl.ArchiveDocumentExtension.Response().error(e.getMessage());
    }
}
}
```

Input parameters of the class:

The input parameter is an object.

```
global class Request {
    global ksl__Document__c record {get;set;}
    global Blob content {get;set;}
    global String contentType {get;set;}
    global String extension {get;set;}
}
```

The record attribute corresponds to the record of the ksl__Document__c being created. Fields such as Id ,

OwnerId , ksl__ProductionDate__c are not populated. The object contains at least the following fields:

The object contains at least the following fields:

Fields	Description
Name	Description entered by the user or in the configuration
ksl__RecordId__c	Source record identifier
ksl__ObjectName__c	API name of source record object
ksl__VersionNumber__c	Version number
ksl__Service__c	KSL service name
ksl__EditMode__c	Edition type. 3 possible values: generate (transactional), edit (interactive), email
ksl__InteractiveDocumentId__c	Identifier of the KSL interactive document

The owner is the current user.

The "content" attribute contains the content of the file.

The "contentType" attribute contains the "Content-Type" corresponding to the file.

The "extension" attribute contains the file extension.

Class output parameters:

```
global class Response {
    private Boolean isSuccess;
    private String documentId;
    private String error;
    global Response success(String documentId) {
        this.isSuccess = true;
        this.documentId = documentId;
        return this;
    }
    global Response error(String error) {
        this.isSuccess = false;
        this.error = error;
        return this;
    }
}
```

Use the "success" method to indicate the success of the operation and the document identifier.

Use the "error" method to indicate a failure.

Running the class:

The call is made via the principle defined in the following link:

```
https://developer.salesforce.com/docs/atlas.en-us.apexref.meta/apexref/apex\_methods\_system\_type.htm#apex\_methods\_system\_type
```

En cas d'erreur sans message, le connecteur utilise un message générique.

In the event of an error without a message, the connector uses a generic message. The ID of the received document is stored in the `ksl__DocumentId__c` field of the `ksl__Document__c` record.

8.3. Implementing APEX classes

Two new apex classes must be implemented for using an external storage system:

- A specific Apex class to drive the sending of the document to the external system, to be developed by the user using the technical description above
- The mandatory unit test class, which allows to test and validate the specific class to pass it in production.

The specific APEX class:

The specific class must be created as a new APEX class in the relevant Salesforce instance.

Example content of a specific Apex class named `SampleKSLArchiveDocumentExtension` that would upload documents to a KSL server used as external storage:

```

/**
 * @description this class must be global in order to be used by the package
 */

    global with sharing class SampleKSLArchiveDocumentExtension implements
ksl.ArchiveDocumentExtension.Process {

global ksl.ArchiveDocumentExtension.Response
process(ksl.ArchiveDocumentExtension.Request request) {
try{
    HttpRequest httpRequest = new HttpRequest();
    httpRequest.setEndpoint('https://nom_du_serveur/webservices/runService?kslUser=
nom_utilisateur&kslPassword=mot_de_passe&kslProjectName=nom_du_projet&kslService=archi
ve_document&FichierArchive=HttpBody&TypeDocument=Test_SF&TypeFichierArchive='+request.
extension);
    httpRequest.setMethod('POST');
    httpRequest.setTimeout(120000);
    httpRequest.setBodyAsBlob(request.content);
    httpRequest.setHeader('Content-Type', request.contentType);
    httpRequest.setHeader('Content-Length', String.valueOf(request.content.size()));
    HttpResponse httpResponse = new Http().send(httpRequest);
    switch on httpResponse.getStatusCode() {
        when 200 {
            System.Pattern pattern = System.Pattern.compile('<VALUE>([^\
><]*)<\/VALUE>');
            System.Matcher matcher =
            pattern.matcher(httpResponse.getBody());
            while(matcher.find()){
                return new
                ksl.ArchiveDocumentExtension.Response().success(matcher.group(1));
            }
            return new ksl.ArchiveDocumentExtension.Response().error('Erreur d\'archivage en GED');
        }
        when else {
            return new ksl.ArchiveDocumentExtension.Response().error('Erreur
d\'archivage en GED');
        }
    } catch(Exception e){
        System.debug(LoggingLevel.ERROR, e.getMessage());
        return new ksl.ArchiveDocumentExtension.Response().error(e.getMessage());
    }
}
}
}

```

La classe Apex de tests unitaires :

The unit testing APEX class is used to test and validate the code coverage of the specific class, a mandatory procedure before going into production. It must be created as a new APEX class in the relevant Salesforce

instance.

Example code for an Apex class named `SampleKSLArchiveDocumentExtensionTest` that can be used to test the specific `SampleKSLArchiveDocumentExtension` class example given above:


```

@Test
public with sharing class SampleKSLArchiveDocumentExtensionTest {
    @IsTest
    private static void testNullParameter(){
        ksl.ArchiveDocumentExtension.Response response = new
        SampleKSLArchiveDocumentExtension().process(null);
        System.assert(!response.isSuccess(), 'Should have failed due to null parameter');
    }
    @IsTest
    private static void testSuccess(){
        ksl.ArchiveDocumentExtension.Request request = new
        ksl.ArchiveDocumentExtension.Request();
        request.extension = 'pdf';
        request.contentType = 'application/pdf';
        request.content = Blob.valueOf('content');
        Test.setMock(HttpCalloutMock.class, new SuccessMock());
        ksl.ArchiveDocumentExtension.Response response = new
        SampleKSLArchiveDocumentExtension().process(request);
        System.assert(response.isSuccess(), 'Should have succeeded');
        System.assertEquals('DOCUMENTID', response.getDocumentId());
    }
    @IsTest
    private static void testFailure(){
        ksl.ArchiveDocumentExtension.Request request = new
        ksl.ArchiveDocumentExtension.Request();
        request.extension = 'pdf';
        request.contentType = 'application/pdf';
        request.content = Blob.valueOf('content');
        Test.setMock(HttpCalloutMock.class, new FailureMock());
        ksl.ArchiveDocumentExtension.Response response = new
        SampleKSLArchiveDocumentExtension().process(request);
        System.assert(!response.isSuccess(), 'Should have failed');
    }
    private class FailureMock implements HttpCalloutMock {
        public HTTPResponse respond(HTTPRequest req) {
            HTTPResponse res = new HTTPResponse();
            res.setHeader('Content-Type', 'application/xml');
            res.setBody('<ERROR>400</ERROR>');
            res.setStatusCode(400);
            return res;
        }
    }
    private class SuccessMock implements HttpCalloutMock {
        public HTTPResponse respond(HTTPRequest req) {
            HTTPResponse res = new HTTPResponse();
            res.setHeader('Content-Type', 'application/xml');
            res.setBody('<VALUE>DOCUMENTID</VALUE>');
            res.setStatusCode(200);
            return res;
        }
    }
}

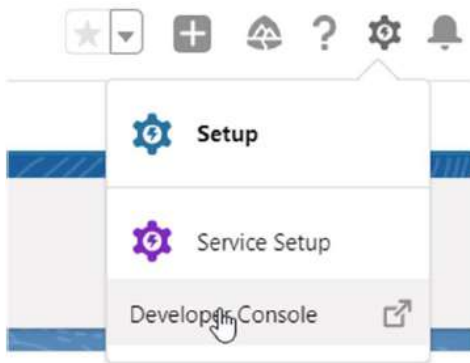
```

```
}  
}
```

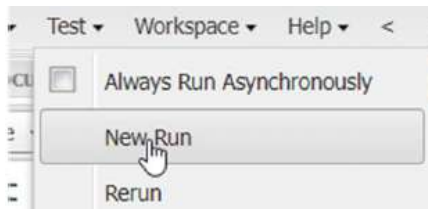
Specific Apex Class Validation:

Steps to validate specific class using unit test class before release, example for SampleKSLArchiveDocumentExtensionTest class:

1. Opening the Developer Console:

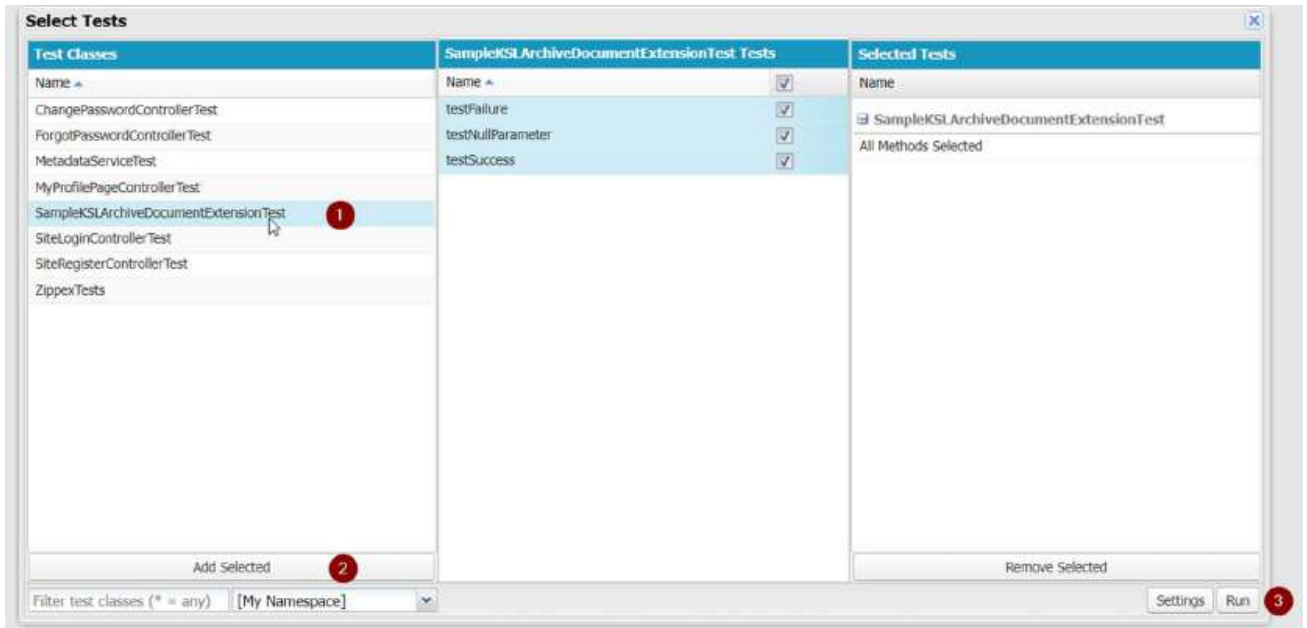


2. Click on the "New Run" button on the "Test" tab



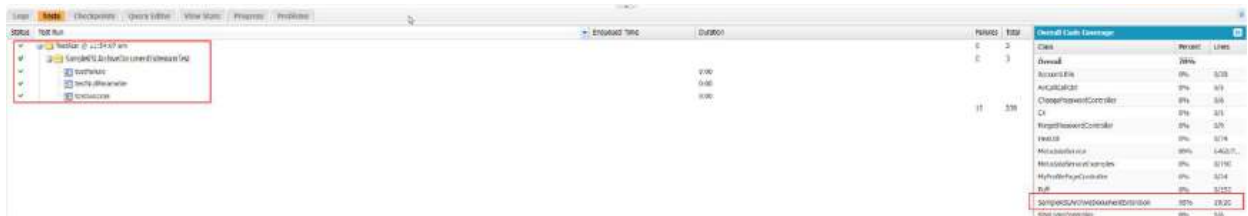
3. Launching the test:

- Select the SampleKSLArchiveDocumentExtensionTest class
- Click on the "Add Selected" button
- Click on the "Run" button



4. Checking the result:

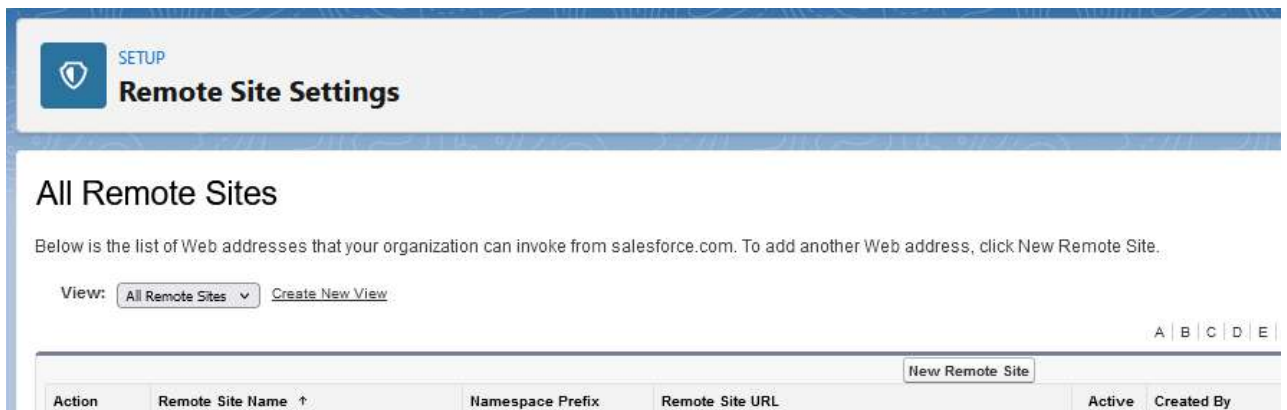
- Open the "Tests" tab (at the bottom)
- Wait for a green or red tick (green tick indicates success)
- Check that the percentage table on the right indicates at least 75% for the class tested, mandatory minimum for production.



8.4. Setting permissions

A new remote site must be set up to allow the Salesforce instance to access the external storage system:

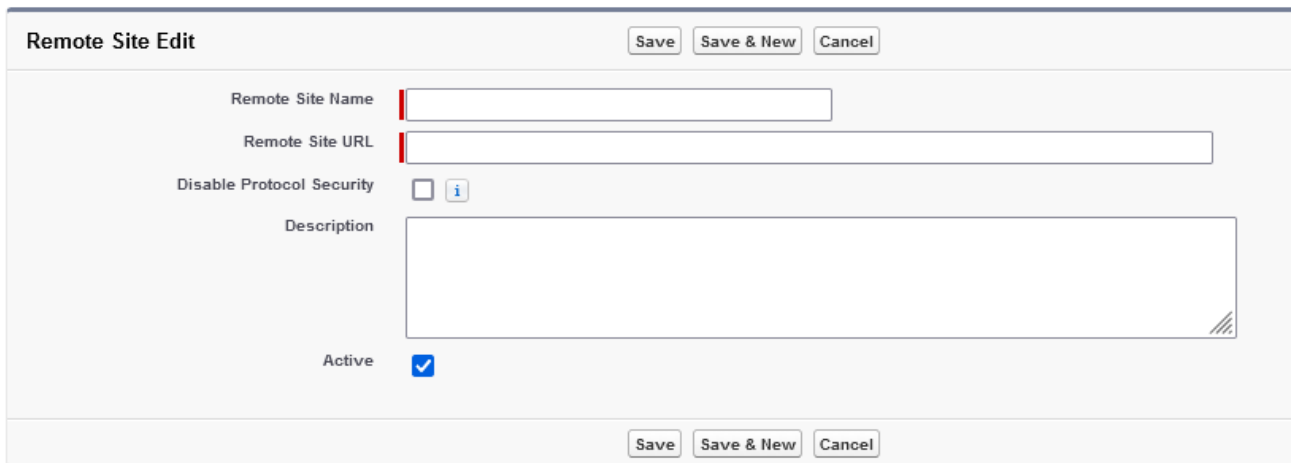
- In Setup , find the Remote Site Settings section .
- Create a new remote site with the New remote site button.



- Specify the following parameters
 - Nom du site distant : nom pour le site distant (exemple : Site stockage externe)
 - Url du site distant : URL d'accès au système de stockage externe (exemple : https://hostname:XXX avec XXX le numéro du port https)
 - Remote site name: name for the remote site (example: External storage site)
 - Remote site url: External storage system access URL (example: https://hostname:XXX with XXX the https port number)

Remote Site Edit

Enter the URL for the remote site. All s-controls, JavaScript OnClick commands in custom buttons, Apex, and AJAX proxy calls can access this Web ac



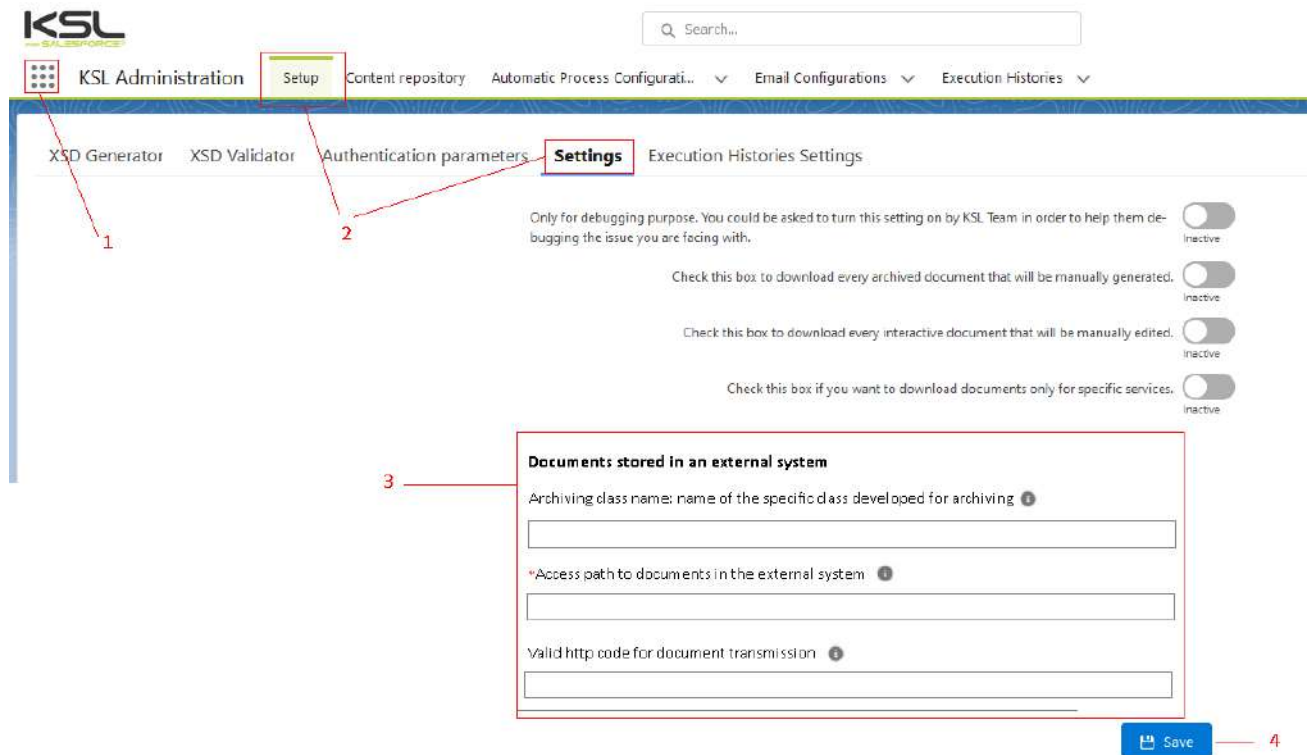
The screenshot shows a web form titled "Remote Site Edit". At the top right, there are three buttons: "Save", "Save & New", and "Cancel". The form contains the following fields and controls:

- Remote Site Name**: A text input field.
- Remote Site URL**: A text input field.
- Disable Protocol Security**: A checkbox that is currently unchecked, with an information icon (i) to its right.
- Description**: A large text area for entering a description.
- Active**: A checkbox that is currently checked.

At the bottom of the form, there are three buttons: "Save", "Save & New", and "Cancel".

8.5. Configuration for external storage

Setting up the KSL For Salesforce plugin for storing documents in the external storage system:



1. Open the KSL For Salesforce plugin configuration by searching for “KSL Administration” in the application menu
2. Go to the “Settings and configuration” tab then “Settings”
3. Specify the following parameters:
 - Archiving class name: name of the specific class developed for archiving (example SampleKSLArchiveDocumentExtension)
 - Access path to documents in the external system: document extraction url in the external storage system, must contain as a parameter the value of the unique identifier of the document using the syntax `{documentId}`, example `https://server_name/webservices?kslService=extract_document&DocumentArchive={documentId}`
4. Save the configuration

9. Annexes

9.1. Generate public and private keys

The Public Key is the identifier of the Ksl Office web application for authorization servers; it is the clientId configured for the KSL Suite application server (web.xml of the Tomcat configuration). This key is dependent on the instance of the KSL Suite server. The public key is required for the KSL server to recognize the Salesforce® server and to allow communication.

The Private Key is the equivalent of the clientSecret, also configured for the KSL Suite application server. This key is required for recognizing the JWT token (JSON Web Token). This 32-byte (256-bit) key corresponds to the "256" of the SHA256 protocol.

a. Private key creation

To generate this private key, we advise you to use the online Jamiekurtz tool: <http://jwtbuilder.jamiekurtz.com> (Signed JSON Web Token part - "Generate 32-byte Key" option).

Be careful, do not click on the Create Signed JWT #button, but on the arrow and select Generate 32-byte Key.

The value of the Key field displayed is the private key that can be copied and pasted into the KSL for Salesforce® configuration.

Previous form information is not needed to generate this key.



b. Public key creation

To generate this public key, we advise you to use the GuidGen online tool: <https://www.guidgen.com>.

Several other ways can be used to create this key:

- From Salesforce®
- With a tool like <https://www.ssh.com/ssh/putty/windows/puttygen>
- With a tool like <http://www.unit-conversion.info/texttools/random-string-generator/>
- With command lines (e.g. openssl)

9.2. Migration to KSL For Salesforce plug-in version 2.2

KSL for Salesforce 2.2 proposes an overhaul of the navigation on the details of the KSL documents, in particular for the interactive document.

If you are working on a version prior to version 2.2 (1.6, 1.8 or 1.8.1), it will be necessary to apply a migration procedure:

- on the history of KSL documents already produced
- on Salesforce objects on which you have installed a KSL document production

On the KSL side, this version requires two upgrades:

- KSL Server upgrade to version 8.1.1.5, mainly for the provision of the new *xml2csv* system service implemented for email
- updating the Salesforce project with the latest level of composite services associated with the Salesforce Plug-In (archiveDocument, email and emailing services if necessary), for the execution of on-demand documents in all formats and the implementation of the *xml2csv* service.

a. Prerequisites for Each Salesforce Object with KSL Models

Do not apply this part on the Opportunity object since its configuration is included in the package.

For each record of the active "KidObjets Description" custom metadata type, you must add a "lookup relationship" field in the KSL Document object (*ksl__Document__c*).

This field should reference the Salesforce object from which you are using the KSL connector.

For example, the "KidObject Opportunity" record, allowing the use of the connector on the Opportunity object, resulted in the creation of a "lookup relationship" field referencing the Opportunity object.

In Setup > Object manager,

Select KSL Document object (*ksl__Document__c*), then Fields and Relationships.

- Click New
- Step 1: On the choice of the type of field, tick Lookup Relationship > Next
- Step 2: On the selection of the related object, select the Salesforce object you want to migrate > Next
- Step 3: Enter the label and the name of the field (*KSLForObjectName* for example) > Next
- Step 4: select the field-level security - profiles list proposed by default > Next
- Step 5: The KSL Document Layout which lists the KSL documents is already ticked in the reference fields for the presentation of the page > Next
- Step 6: Adding personalized associated lists - the object page layout is already checked; click on Save.

b. The migration script on the history of KSL documents

Note that :

- Old recordings will not be deleted
- Each execution of the script works on all the old recordings (several executions without first deleting will therefore create duplicates)

When all the "Lookup Relationship" fields are available for the targeted objects, open the developer console:

Setup > Developer Console

On the menu bar, click on Debug then Open Execute Anonymous Window.

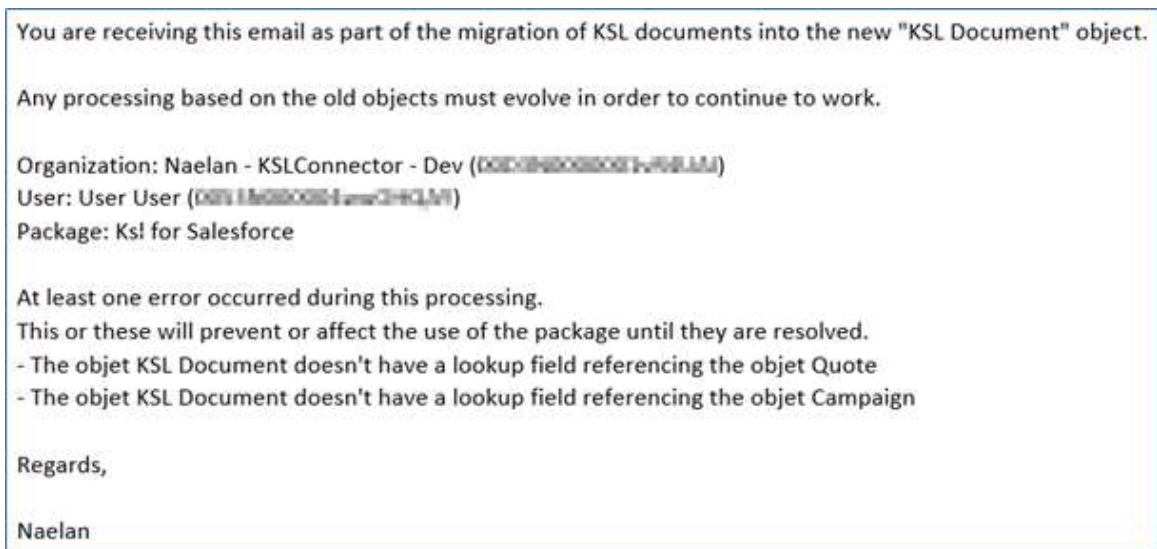
In the opened window, copy / paste the code below:

```
Database.executeBatch(new ksl.PostInstall_MigrationRecordBatch());
```

Click Execute

A batch processing is started on the history of your KSL documents. You will receive an Email as soon as the processing is complete. It will tell you if an error has occurred or not.

For example :



There was no error on the records, but I am told that the KSL Document object does not have a lookup relationship field for the Quote and Campaign objects. Since the connector is ultimately not used on these objects in the context of the example, these two messages are irrelevant.

c. Embed the New KSL Document List on Your Salesforce Objects

To be performed on each Salesforce object with active KSL models.

This new KSL document list should replace the old one on the object page. Following the passage of the script, it takes all the KSL documents generated for the object, old and new.

In Setup > Object Manager

Select the Salesforce object you want to migrate, then Page Layouts.

Click the name of the available page layout.

On the detail of the presentation,

- Select Related Lists from the list on the left
- Among the lists, select the KSL Documents box which is not grayed out and drag it onto the page, under the already existing KSL Documents block

The screenshot displays the Salesforce Page Layout Editor interface. The 'Related Lists' section is visible, containing three blocks:

- KSL Documents (Abbreviated List):** A table with columns: Document description, Owner First Name, Owner Last Name, Created Date, and Version number. A sample row shows 'Sample Text' for description, 'Sample Text' for owner names, '7/29/2021, 4:40 PM' for created date, and '471' for version number.
- KSL Documents (Detailed List):** A table with columns: Name, Created Date, Owner First Name, Owner Last Name, Version Number, and Description. A sample row shows 'Sample Text' for name and description, '7/29/2021, 4:40 PM' for created date, 'Sample Text' for owner names, and '67,718' for version number.
- Open Activities:** A table with columns: Subject, Name, Task, Due Date, Status, Priority, and Assigned To. A sample row shows 'Sample Text' for subject and name, a checkmark for task, '7/29/2021, 4:40 PM' for due date, 'Sample Text' for status, 'Sample Text' for priority, and 'Sarah Sample' for assigned to.

- Configure this second KSL Documents block by clicking on the adjustable wrench, and select the fields you want to appear on the list of KSL documents, knowing that the first 4 are those presented on the abbreviated list.

Related List Properties - KSL Documents

Help ?

Columns -

Select fields to display on the related list. You can also re-order the selected fields.

Available Fields

- Compte
- Created By
- Created By Alias
- Current Version
- Document Id
- Edit Mode
- ExternalId
- File Id

Selected Fields

- Name
- Created Date
- Owner First Name
- Owner Last Name
- Version Number
- Description

Sort By: Created Date

Ascending

Descending

Buttons +

OK Cancel Revert to Defaults

- Think about sorting this list of documents, by decreasing creation date to see the most recent documents for example.
- OK
- Delete first block KSL Documents = old version block
- Click Save to save the page changes and exit, or Quick Save to stay on the page.

9.3. FAQ

a. "Bilingual" interface problem on plug-in elements?

The context of the problem is a combinatorial:

- a new Salesforce org or sandbox
- a user whose login language is not English

The consequence is a series of interfaces, especially in KSL Administration but also on the interactive document, partially on the language of the user AND in English.

To solve the problem,

Configuration > User interface > Translation system > Translation language settings

Activate the translation system

-> the two or three languages (from version 2.2) supported are displayed in the list

A screen refresh is sufficient to see a uniform translation appear.



NAELAN

Headquarter - 4 rue Claude Chappe
69370 Saint-Didier au Mont d'Or
France - Tél. +33 4 37 59 81 40

Paris Office - 4 Place Louis Armand
75023 Paris - France
Tél. +33 (0)1 72 76 80 86

www.naelan.com
contact@naelan.com
support@naelan.com