Public Cloud Custom Form Design

30 July 2024 | Document Version 0.01

Berk Saldır

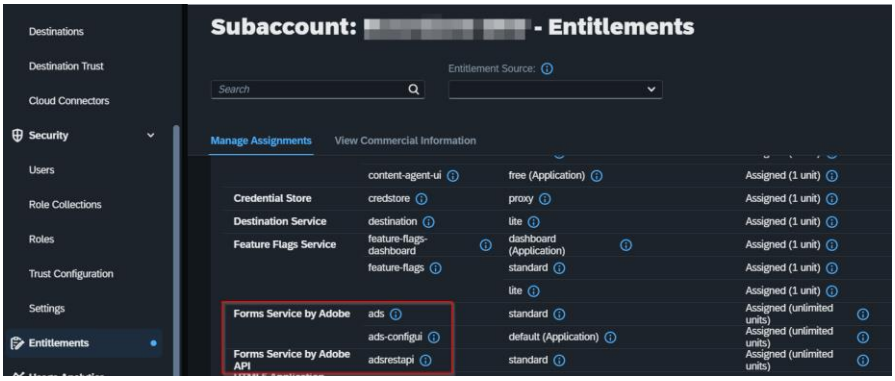
Contents

SAP Forms Service by Adobe

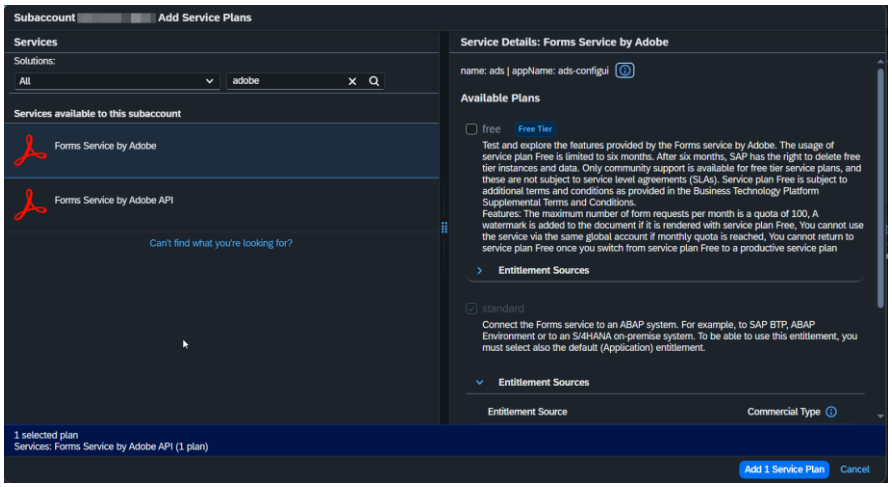
SAP Forms service by Adobe lets you generate print and interactive forms using Adobe Document Services (ADS). Calling the service from cloud application using a REST API for rendering documents and for managing form templates in the template store.

1. Service Adjustments on the BTP Account

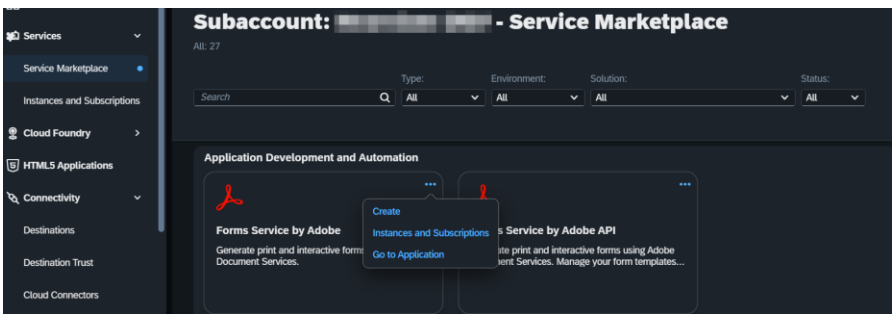
Firstly, it should be checked whether there are services in Entitlements part of subaccount.



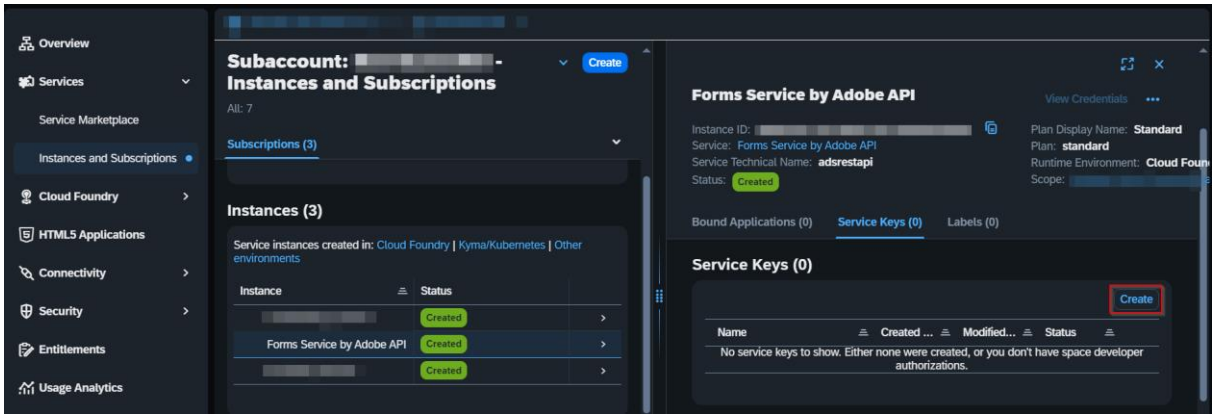
If services are not available, they should be added. To add them, edit mode should be opened and clicked “Add Service Plans” button. The services listed on the pop-up screen must be added.



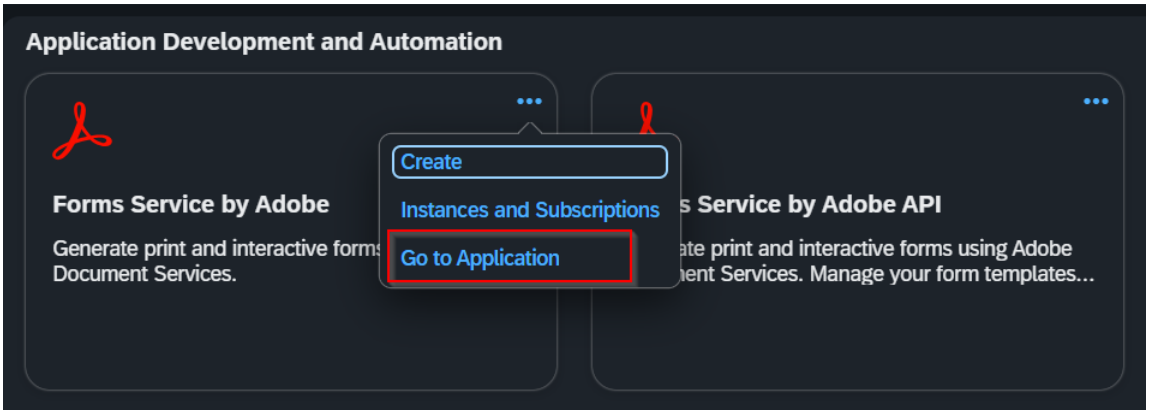
Default plan should be selected for “Forms Service by Adobe” service. Standard plan should be selected for “Forms Service by Adobe API” service. After that part, some adjustments will be made in the Services part of subaccount. Both services need to be created with selected plans.

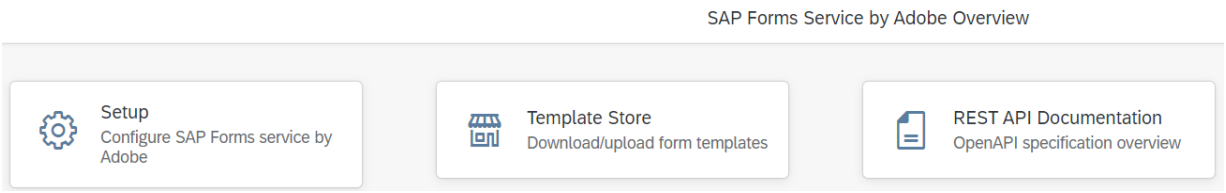


When creating “Forms Service by Adobe API”, “Runtime Environment” will be asked. “Cloud Foundry” should be selected. If “Cloud Foundry” is not enabled, it should be enabled by contacting an authorized person. After the “Forms Service by Adobe API” is created, a service key is generating from “Instances and Subscriptions”.



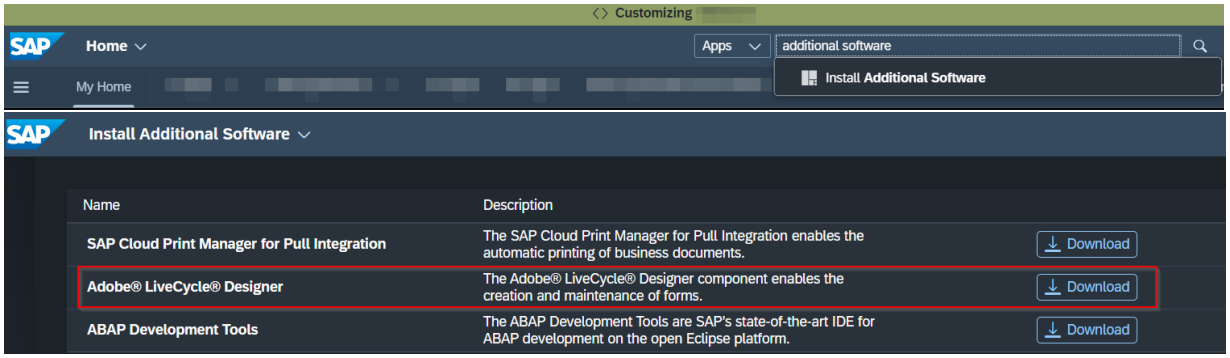
After completing adjustments, an app and a service key is created. The app is used to store form templates and to make configurations of SAP forms service by Adobe. The service key is used for creating custom forms from ABAP codes. The app can be opened as follows.



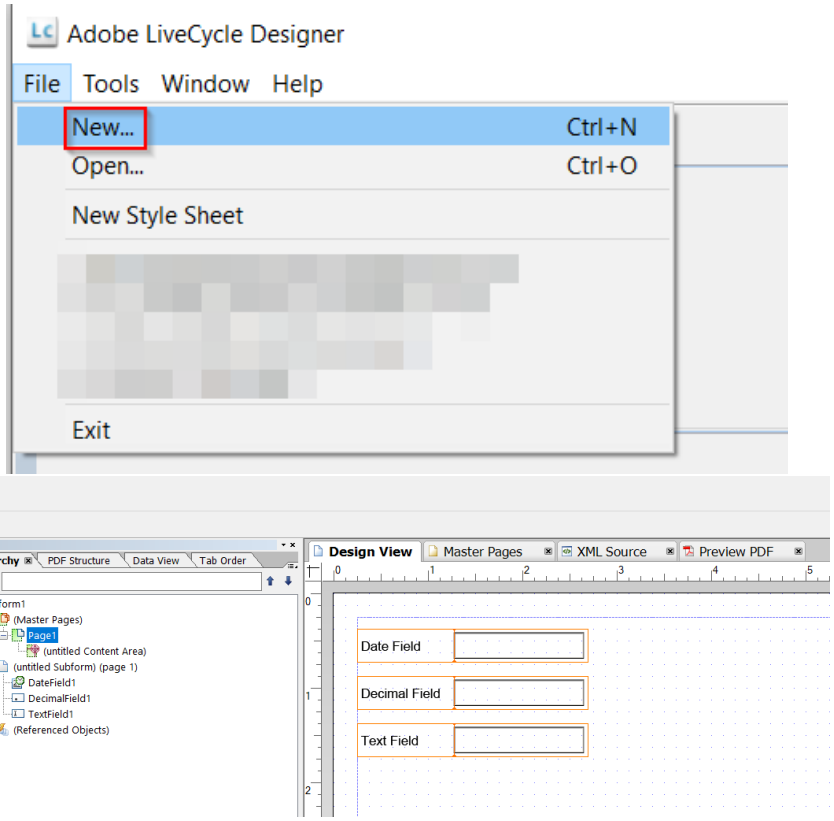


2. Custom Form Creation and Upload

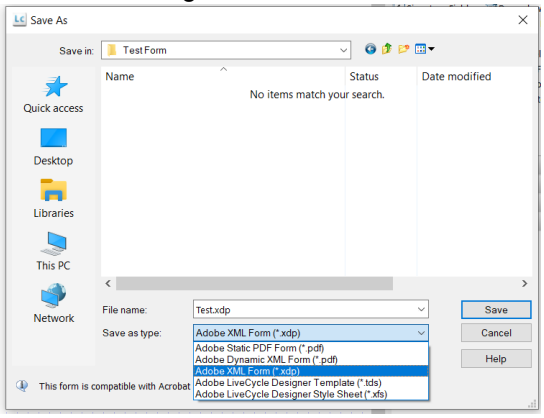
“Adobe LiveCycle Designer” is used to create custom forms. If the designer isn’t installed, it should be installed first. The installation file can be downloaded from the cloud system as follows.



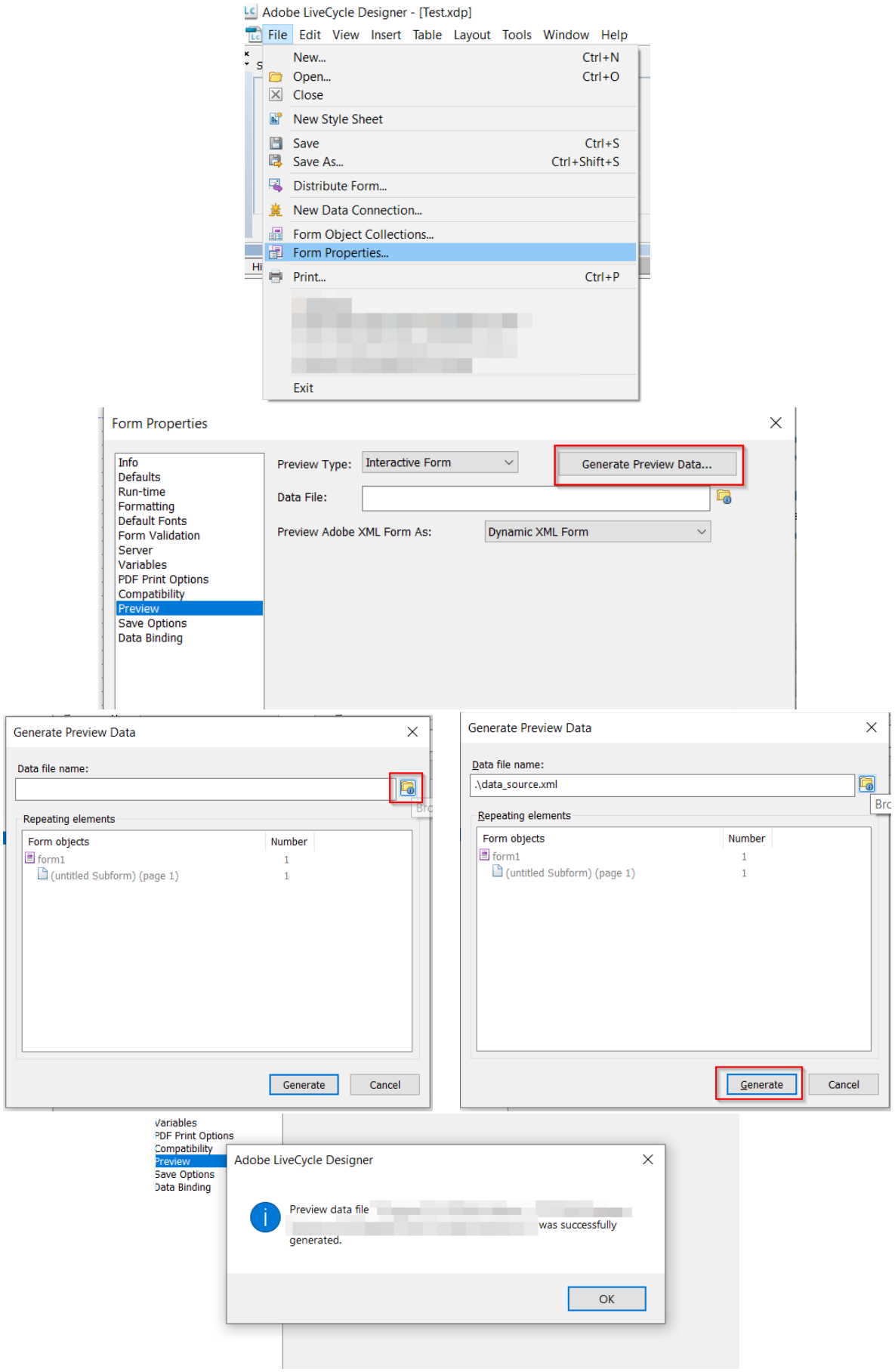
After installation test form can be created for uploading template store. Some fields which are text fields, decimal fields, date fields etc. can be added to test form.

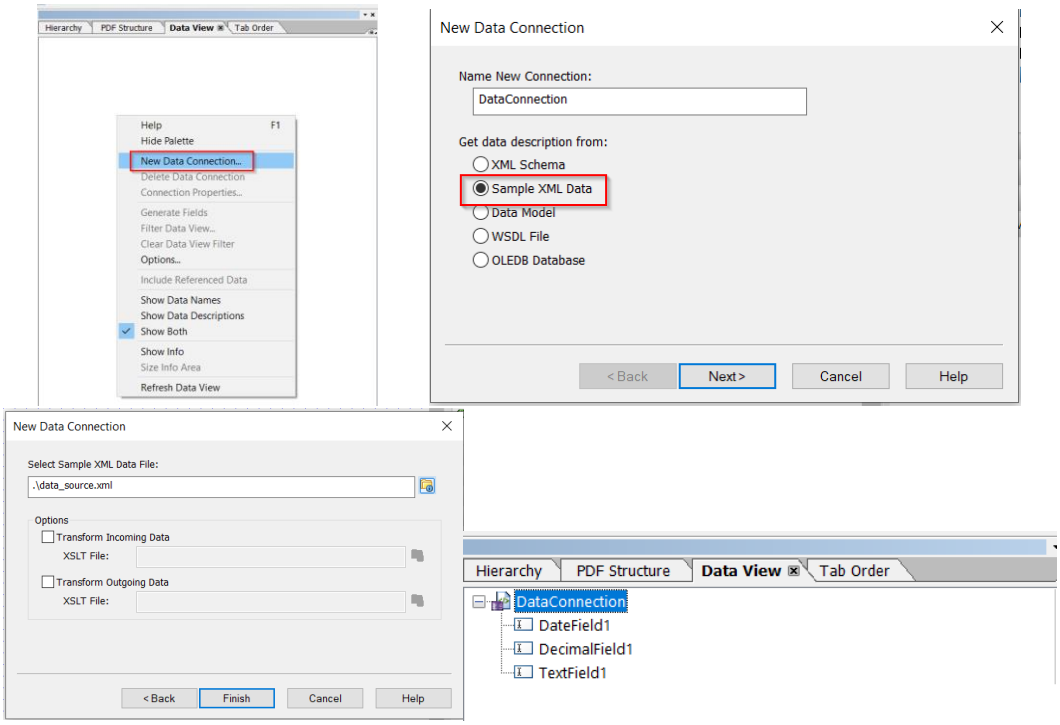


Below selection should be selected when saving test form.

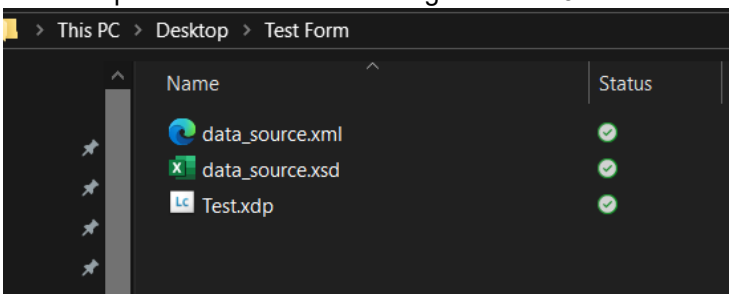


There is a “Data View” tab in left side of designer. Binding values are seen there. Following steps should be followed first for them to be seen.

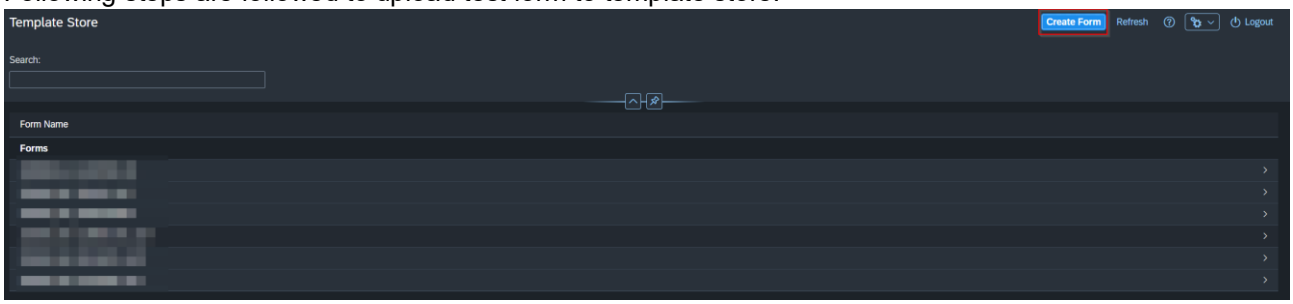


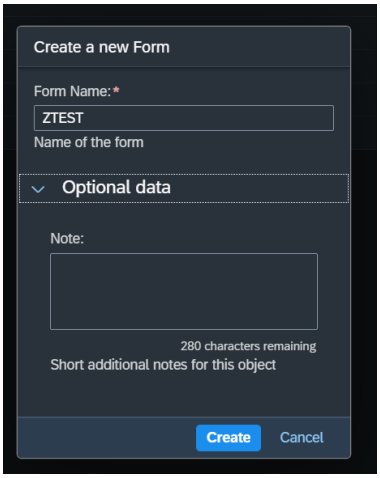


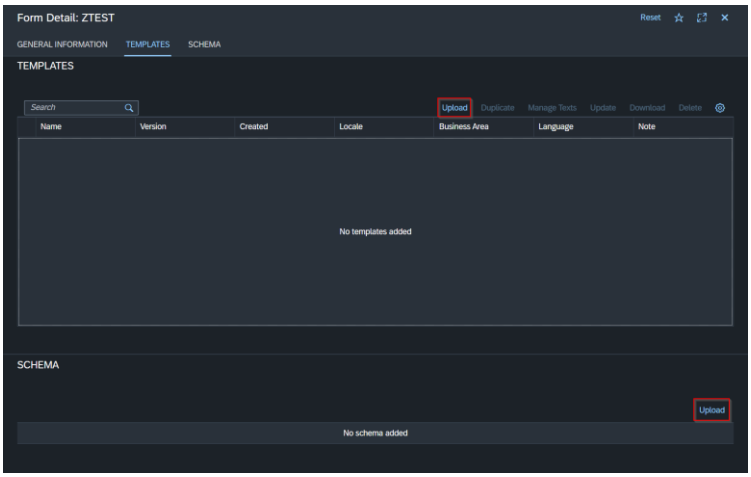
Now, binding values can be used for fields added to the form. XML file should be converted to XSD file for uploading binding schema to template store. After converting there are 3 files in test form folder.

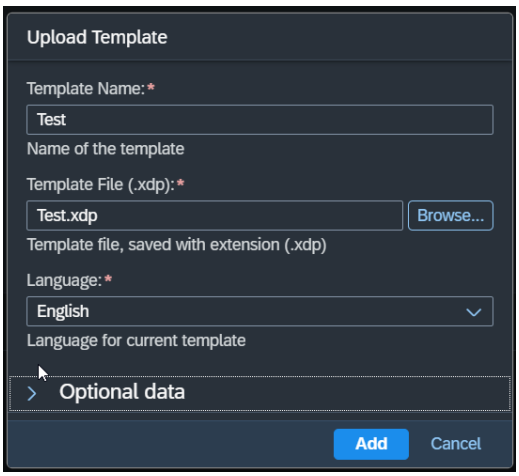


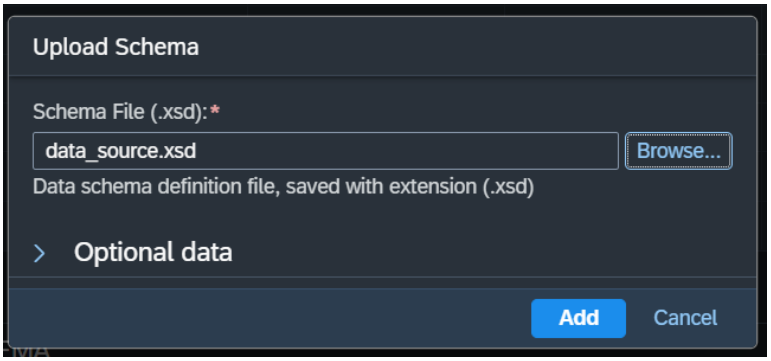
Following steps are followed to upload test form to template store.





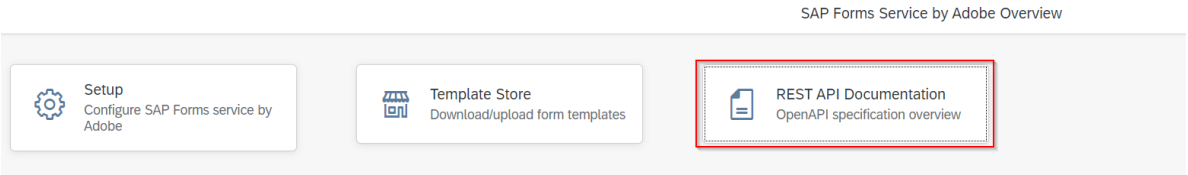




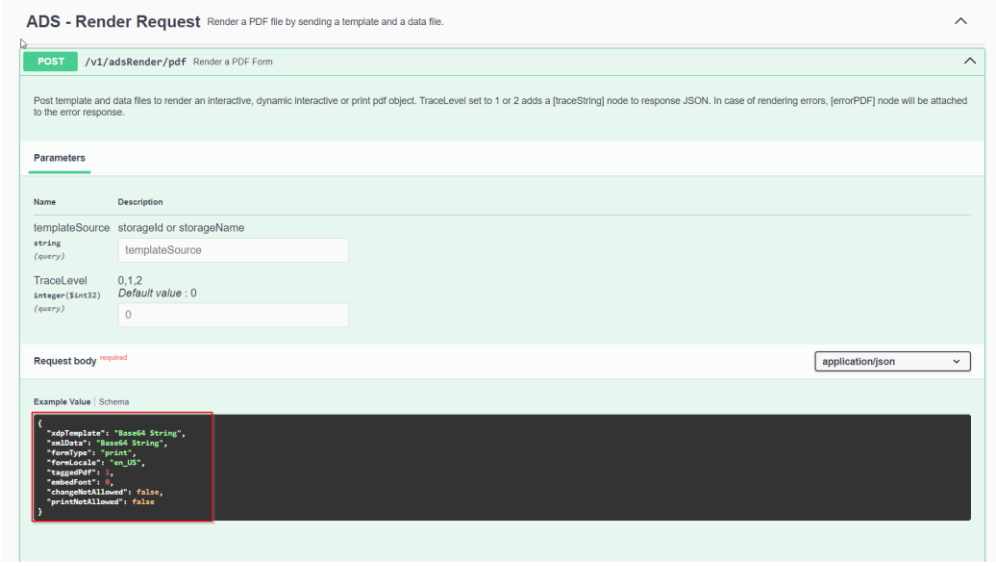


3. Forms Service by Adobe API Configurations in the System

API document is opened using the app.

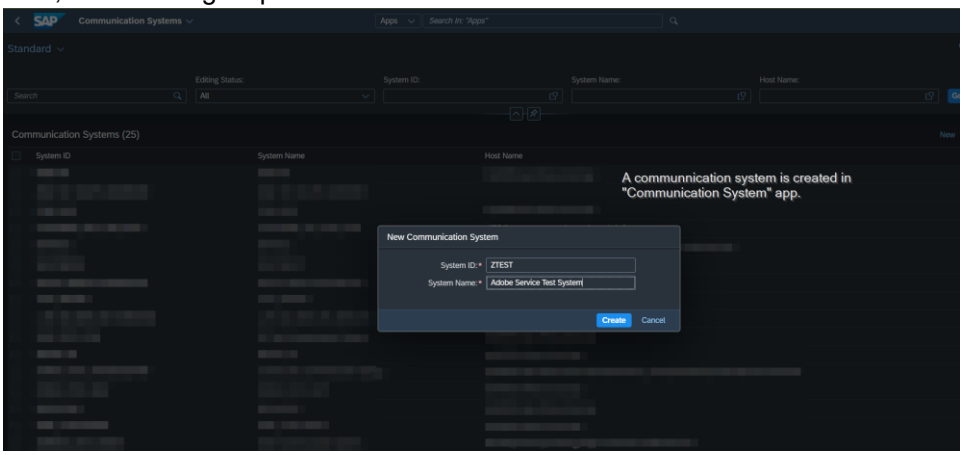


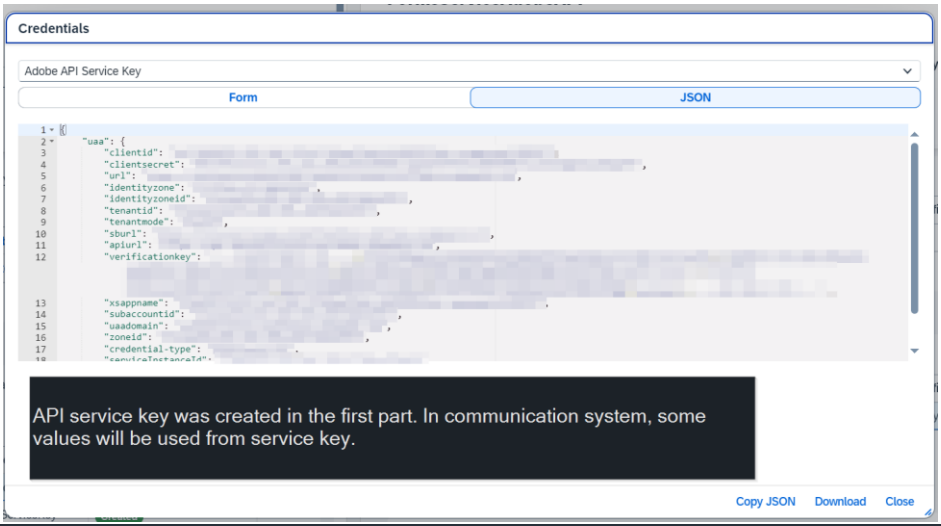
“ADS – Render Request” method is used to render pdf. Method responses can be examined on this page. Inputs of method is as follows.

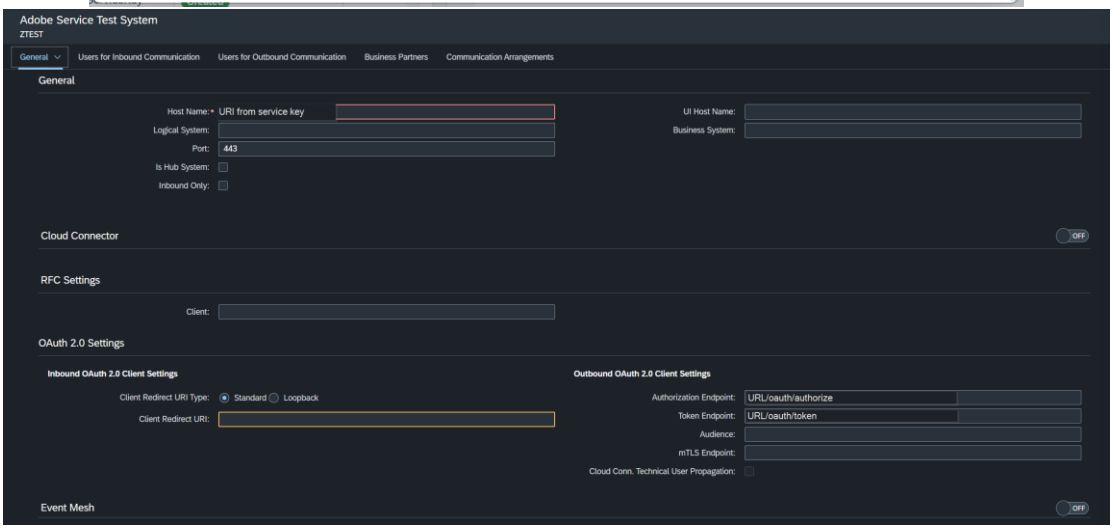


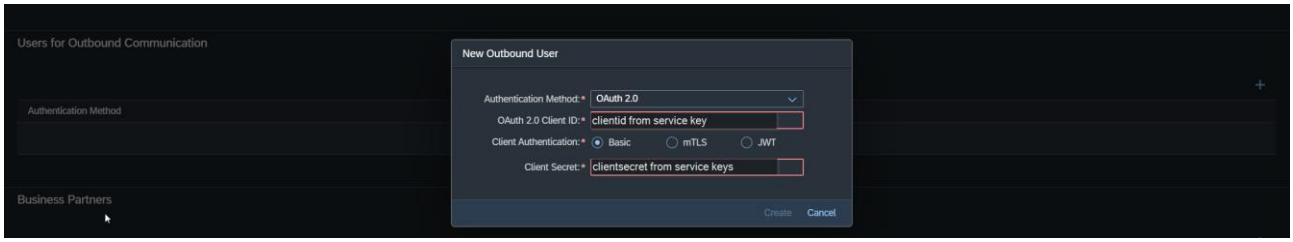
Parameters can be filled with fixed values in the service link. There are 8 input values in the request body. “xdpTemplate” is used for template name. “xmlData” is used to transfer bind values. Type of this field is base64. Bind values should be created in XML format. Then it should be converted to Base64 format in ABAP code. Transformation object can be used to convert a structure to XML. Other input values can be filled in as in the example value.

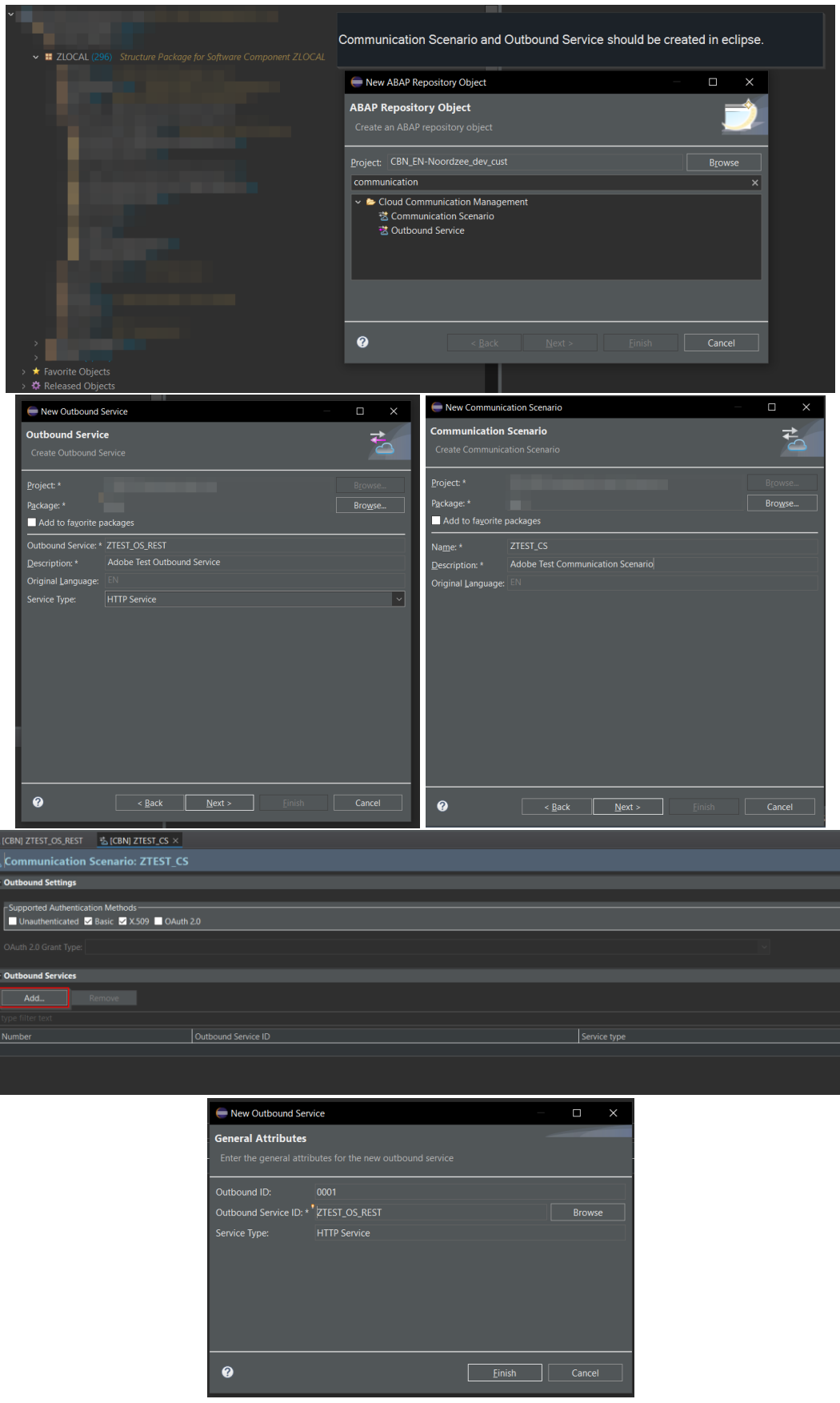
To use the service, the following steps must be made.

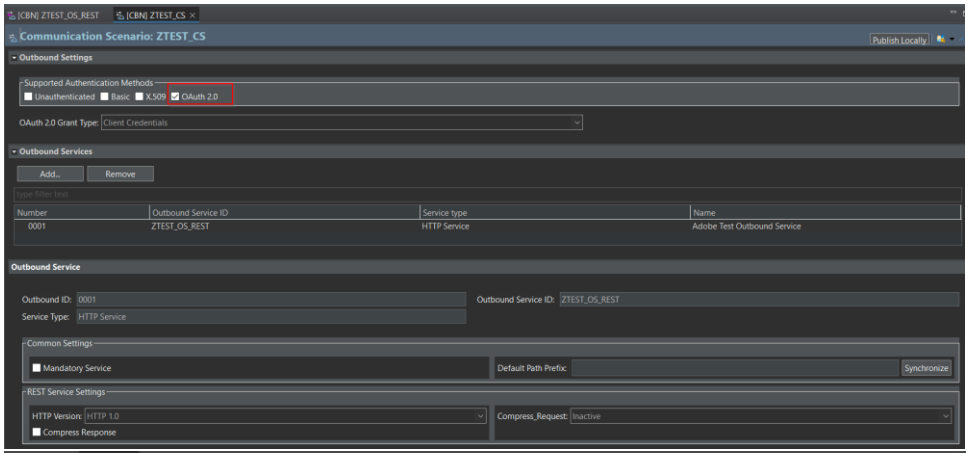


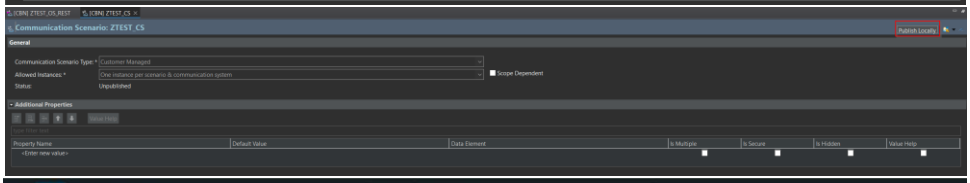


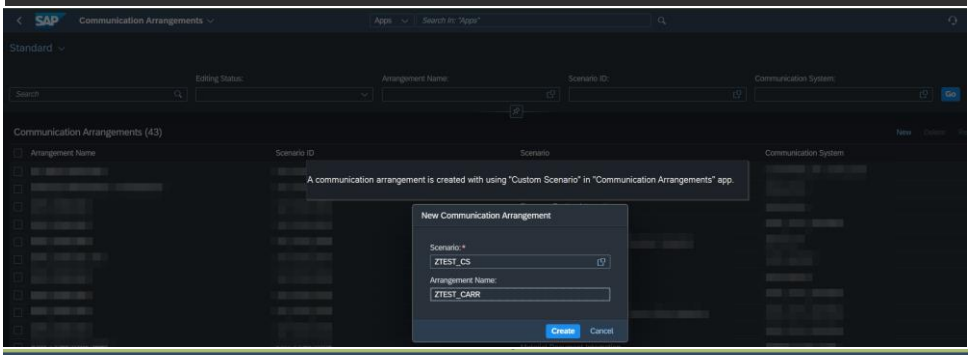


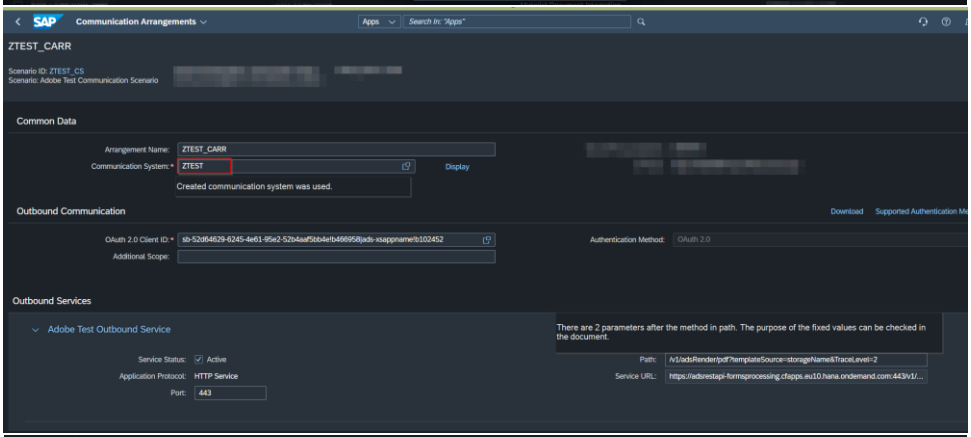


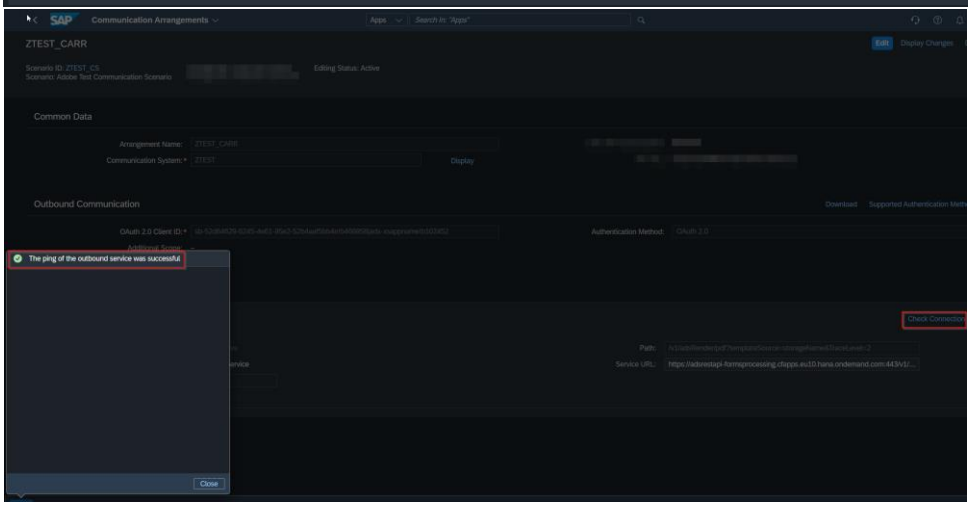



Communication Scenario: ZTEST_CS
Publish Locally
Outbound Settings
Supported Authentication Methods
Unauthenticated
Basic
X.509
OAuth 2.0
OAuth 2.0 Grant Type: Client Credentials
Outbound Services
Add...
Remove
Number
Outbound Service ID
Service type
Name
0001
ZTEST_OS_REST
HTTP Service
Adobe Test Outbound Service
Outbound Service
Outbound ID: 0001
Outbound Service ID: ZTEST_OS_REST
Service Type: HTTP Service
Common Settings
Mandatory Service
Default Path Prefix:
Synchronize
REST Service Settings
HTTP Version: HTTP 1.0
Compress_Request: Inactive
Compress Response


Communication Scenario: ZTEST_CS
Publish Locally
General
Additional Properties


Communication Arrangements
Standard
Communication Arrangements (43)
A communication arrangement is created with using "Custom Scenario" in "Communication Arrangements" app.
New Communication Arrangement
Scenario: *
ZTEST_CS
Arrangement Name:
ZTEST_CARR
Create
Cancel


Communication Arrangements
ZTEST_CARR
Common Data
Arrangement Name: ZTEST_CARR
Communication System: * ZTEST
Display
Created communication system was used.
Outbound Communication
Download
OAuth 2.0 Client ID: *
Authentication Method:
Additional Scope:
Outbound Services
Adobe Test Outbound Service
There are 2 parameters after the method in path. The purpose of the fixed values can be checked in the document.
Service Status: Active
Application Protocol: HTTP Service
Port: 443
Path:
Service URL:


Communication Arrangements
ZTEST_CARR
Edit
Display Changes
Common Data
Outbound Communication
The ping of the outbound service was successful
Close
Check Connection

4. Call Forms Service by Adobe from ABAP Code

Objects should be created before writing code.

ZTEST_S_BINDING

```
@EndUserText.label : 'Binding Structure'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
define structure ztest_s_binding {

  datefield1     : abap.dats;
  decimalfield1  : abap.dec(8,5);
  textfield1     : abap.char(50);

}
```

Structure of data binding is created. That can be changed in every form. It is used in transformation.

```
@EndUserText.label : 'Request Body'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
define structure                  {

  xdp_template       : abap.string(0);
  xml_data           : abap.string(0);
  form_type          : abap.string(0);
  form_locale        : abap.string(0);
  tagged_pdf         : abap.string(0);
  embed_font         : abap.string(0);
  change_not_allowed : abap_boolean;
  print_not_allowed  : abap_boolean;

}
```

This structure will be used for the request body of the method to be used.

```
@EndUserText.label : 'Response Body'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
define structure                  {

  filecontent : abap.string(0);

}
```

Filecontent field is in response body which is returned in response code 200. That structure is used to convert response json body in deserialize method.

[CBN] ZTEST_TR_BINDING

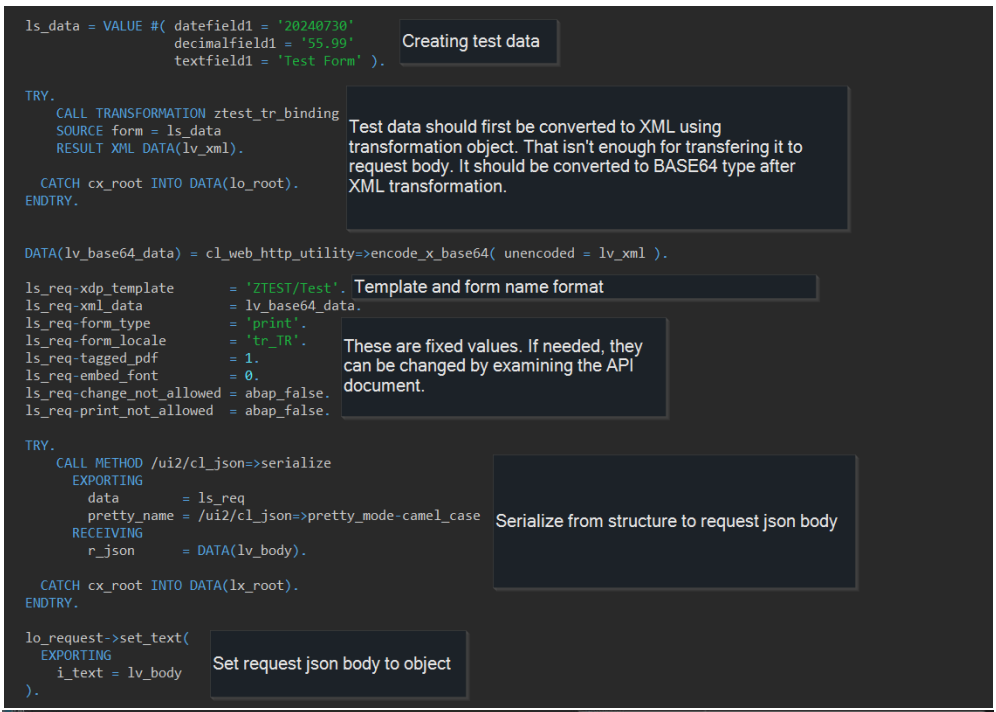
```
<?sap.transform simple?>
<tt:transform xmlns:tt="http://www.sap.com/transformation-templates" xmlns:ddic="http://www.sap.com/abapxml/types/dictionary" xmlns:def="http://www.sap.com/abapxml/types/defined">

<tt:root name="ROOT"/>
<tt:root name="Form"/>
<tt:template>
  <form1>
    <DateField1 tt:value-ref=".Form.DateField1"/>
    <DecimalField1 tt:value-ref=".Form.DecimalField1"/>
    <TextField1 tt:value-ref=".Form.TextField1"/>
  </form1>
</tt:template>

</tt:transform>
```

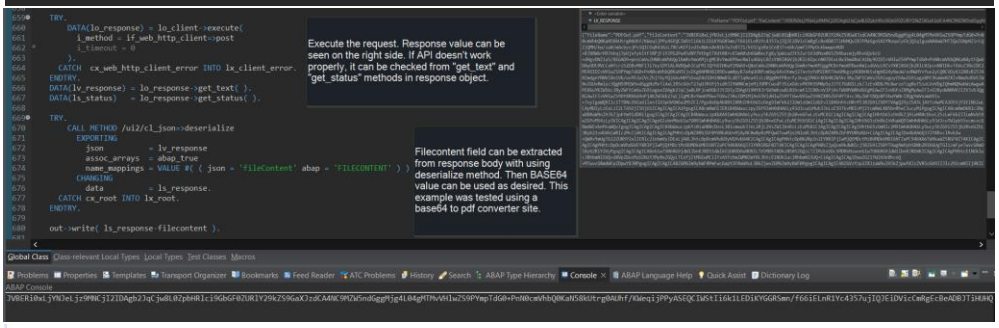
This transformation object is used to convert a binding structure to XML. Content of the transformation object can be changed other forms.

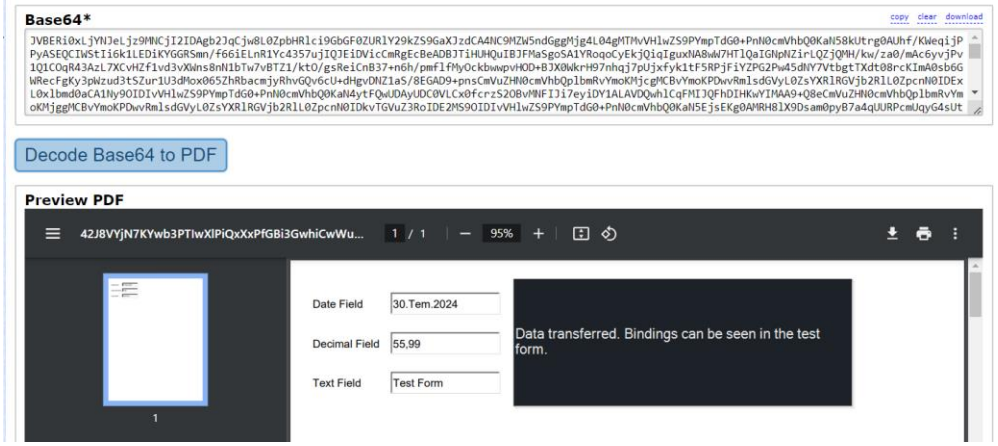
ABAP codes to render PDFs are as follows.

```
TRY.
    DATA(lo_dest) = cl_http_destination_provider=>create_by_comm_arrangement(
      comm_scenario  = 'ZTEST_CS'
      comm_system_id = 'ZTEST'
      service_id     = 'ZTEST_OS'
    ).
  CATCH cx_http_dest_provider_error INTO DATA(lx_error).
ENDTRY.

TRY.
    DATA(lo_client) = cl_web_http_client_manager=>create_by_http_destination( lo_dest ).
  CATCH  cx_web_http_client_error INTO DATA(lx_client_error).
ENDTRY.

DATA(lo_request) = lo_client->get_http_request( ).
lo_request->set_header_fields( VALUE #(
  ( name = 'Accept' value = 'application/json, text/plain, */*'  )
  ( name = 'Content-Type' value = 'application/json;charset=utf-8'  )
) ).
```

HTTP objects should be created. Custom com. scenario, custom outbound service and custom com. system are used when creating destination object.

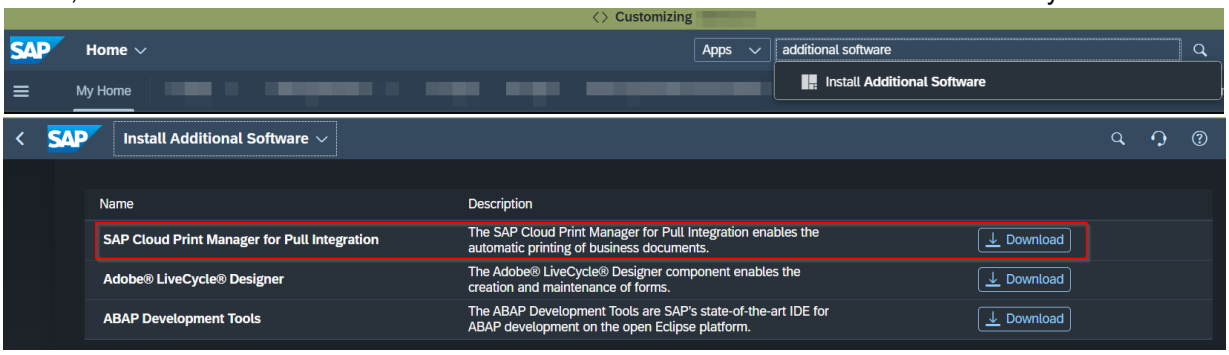

Creating test data
Test data should first be converted to XML using transformation object. That isn't enough for transfering it to request body. It should be converted to BASE64 type after XML transformation.
Template and form name format
These are fixed values. If needed, they can be changed by examining the API document.
Serialize from structure to request json body
Set request json body to object


Execute the request. Response value can be seen on the right side. If API doesn't work properly, it can be checked from "get_text" and "get_status" methods in response object.
Filecontent field can be extracted from response body with using deserialize method. Then BASE64 value can be used as desired. This example was tested using a base64 to pdf converter site.

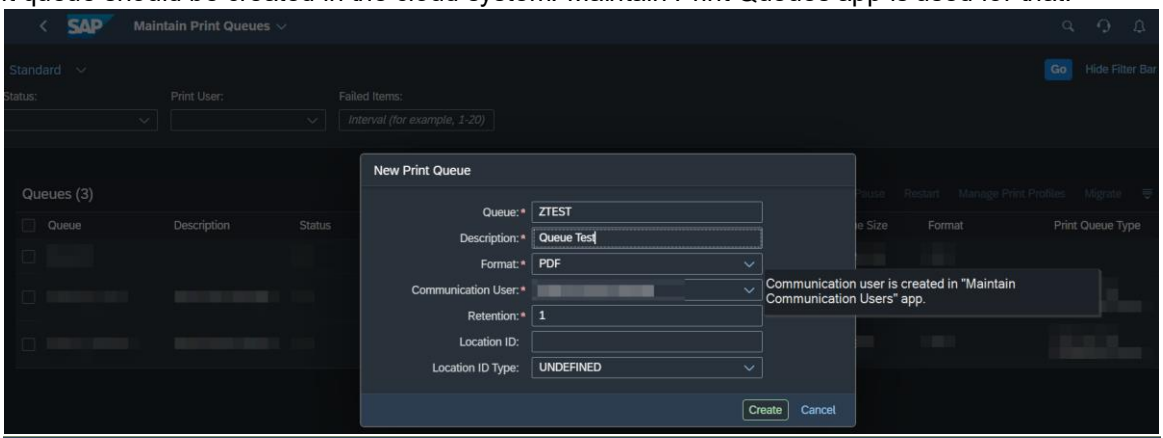

Base64*
copy clear download
Decode Base64 to PDF
Preview PDF
42J8VYjN7KYwb3PTIwXlPiQxXxPfGBi3GwhiCwWu...
Date Field
30.Tem.2024
Decimal Field
55,99
Text Field
Test Form
Data transferred. Bindings can be seen in the test form.

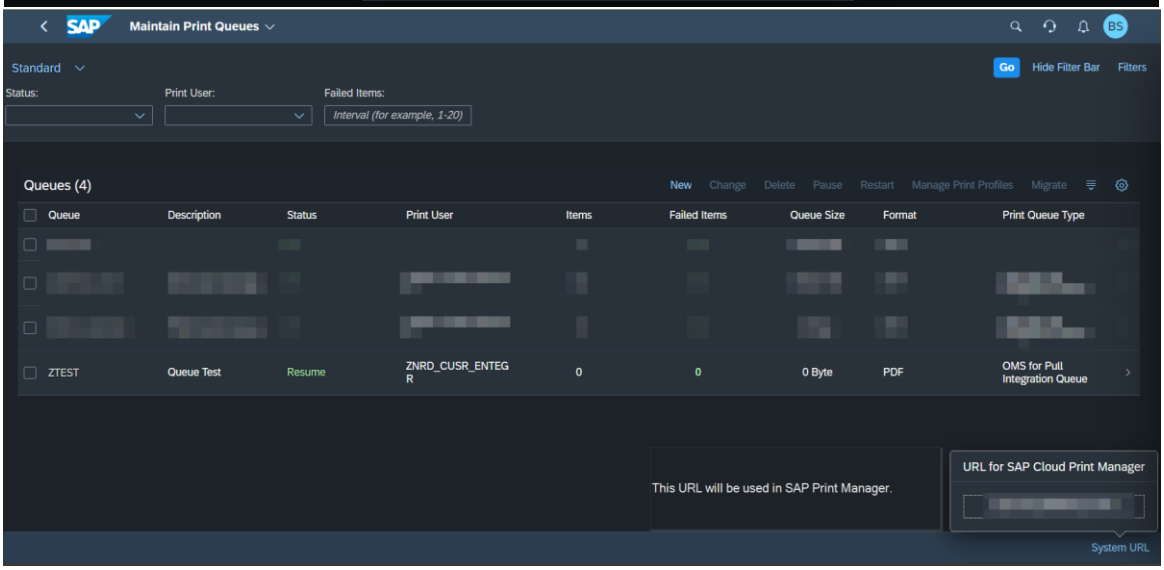
5. Send Rendered PDF to Print Queue

“SAP Cloud Print Manager for Pull Integration” is used to send documents to printer. If the print manager isn’t installed, it should be installed first. The installation file can be downloaded from the cloud system as follows.

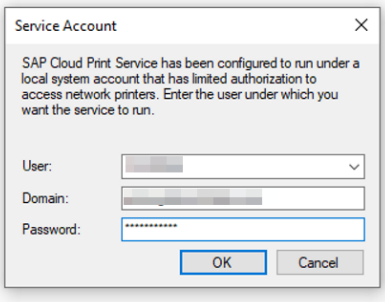


A print queue should be created in the cloud system. Maintain Print Queues app is used for that.

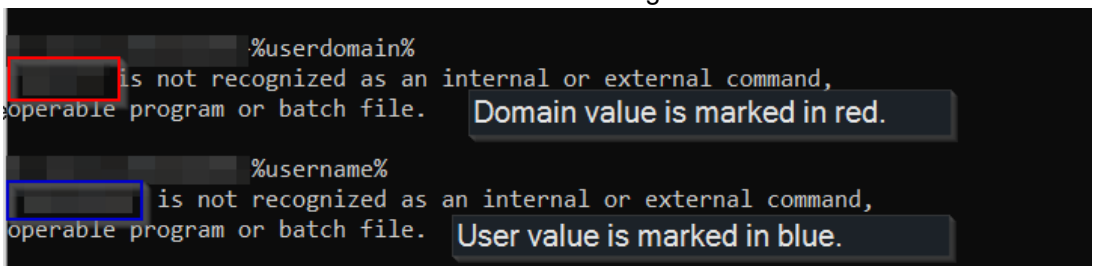




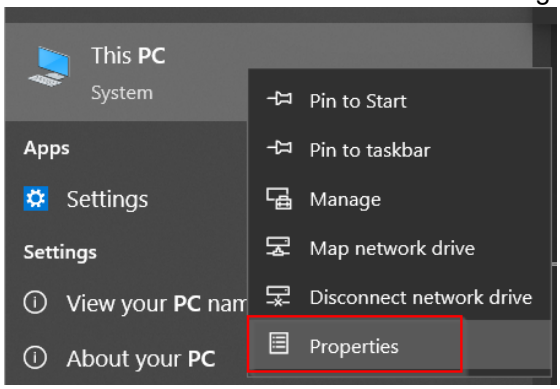
When SAP Print Manager is opened, the following screen appears.

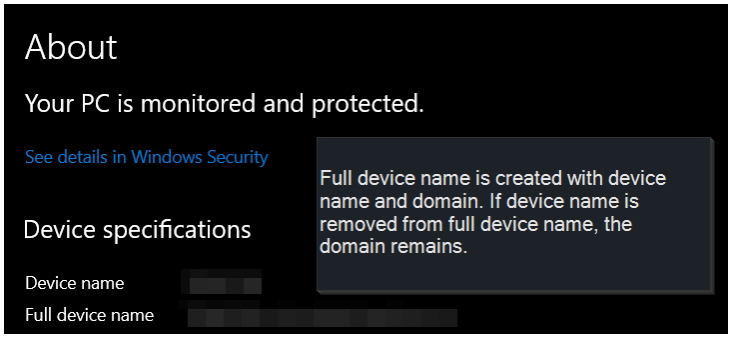


User and domain values can be found in CMD with the following commands.

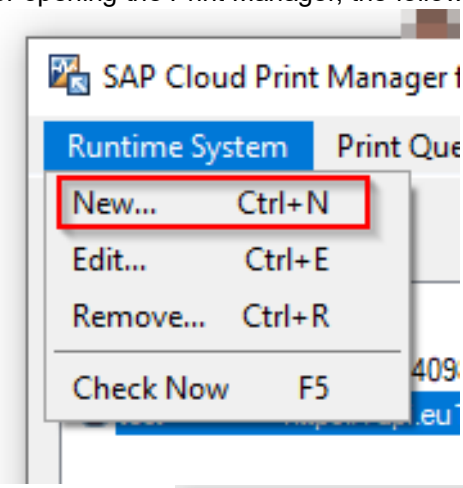


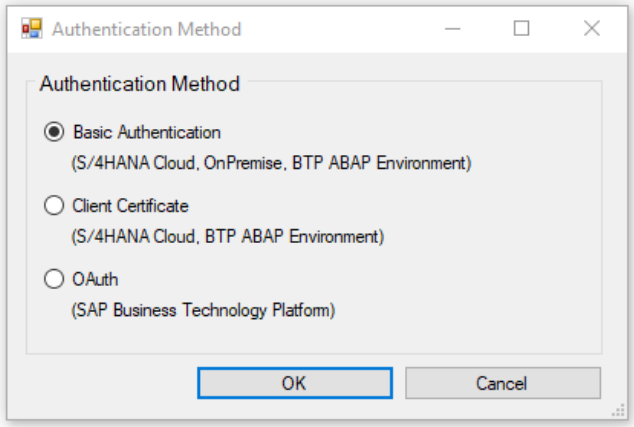
Domain value can also be found with following steps.

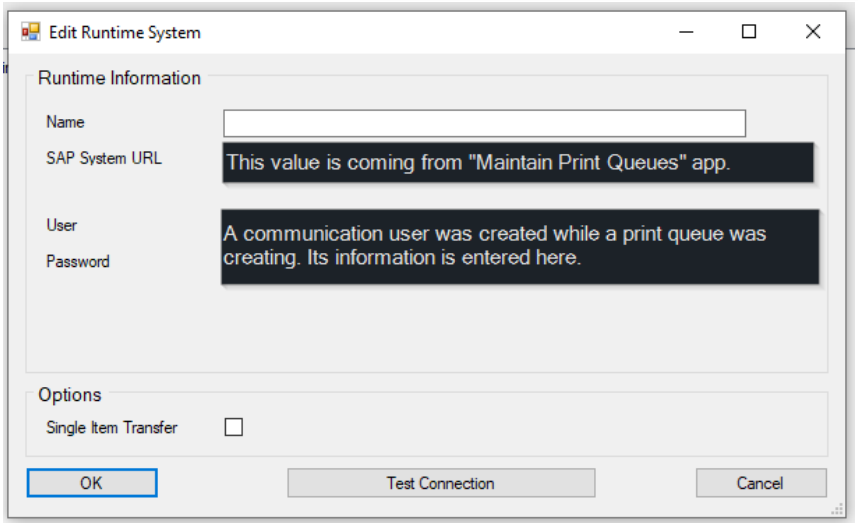


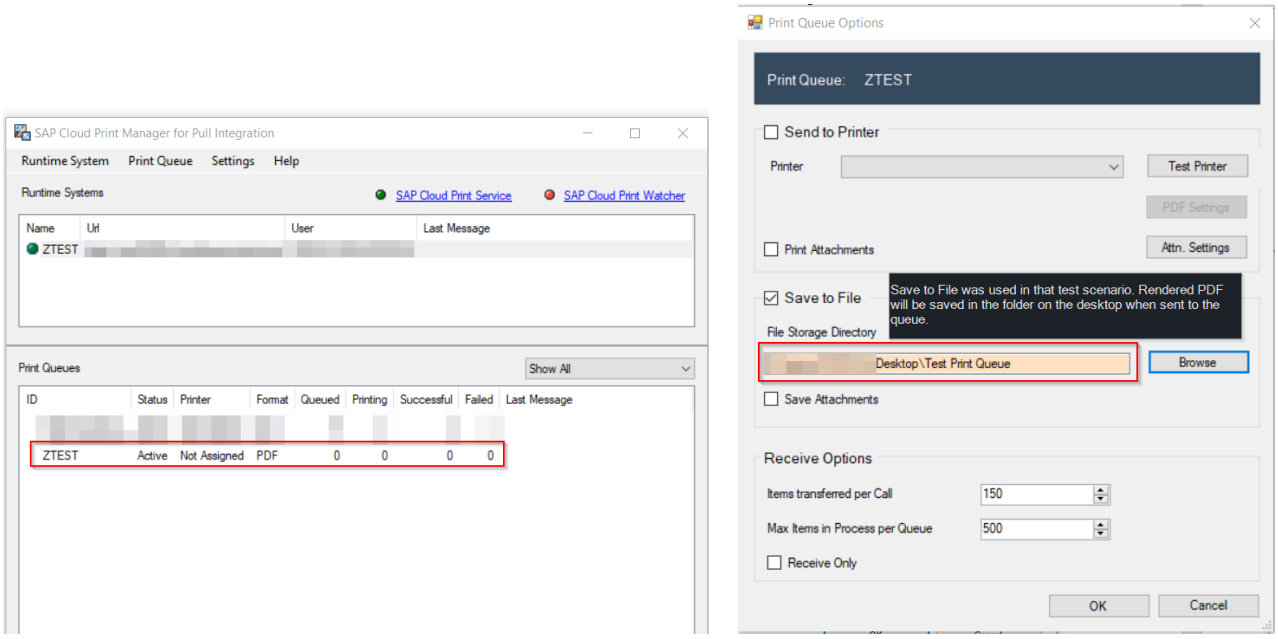


After opening the Print Manager, the following steps should be followed.









That was it for the Print Manager settings. To send a PDF from ABAP code to queue, the following steps are applied.