CRM DOXIS Integration

Configuration manual

26. 09. 2022

Title	CRM DOXIS Integration
Subject	Configuration manual
Customer	SER
Project	SER
Status	Draft
Author	Tom Tajić

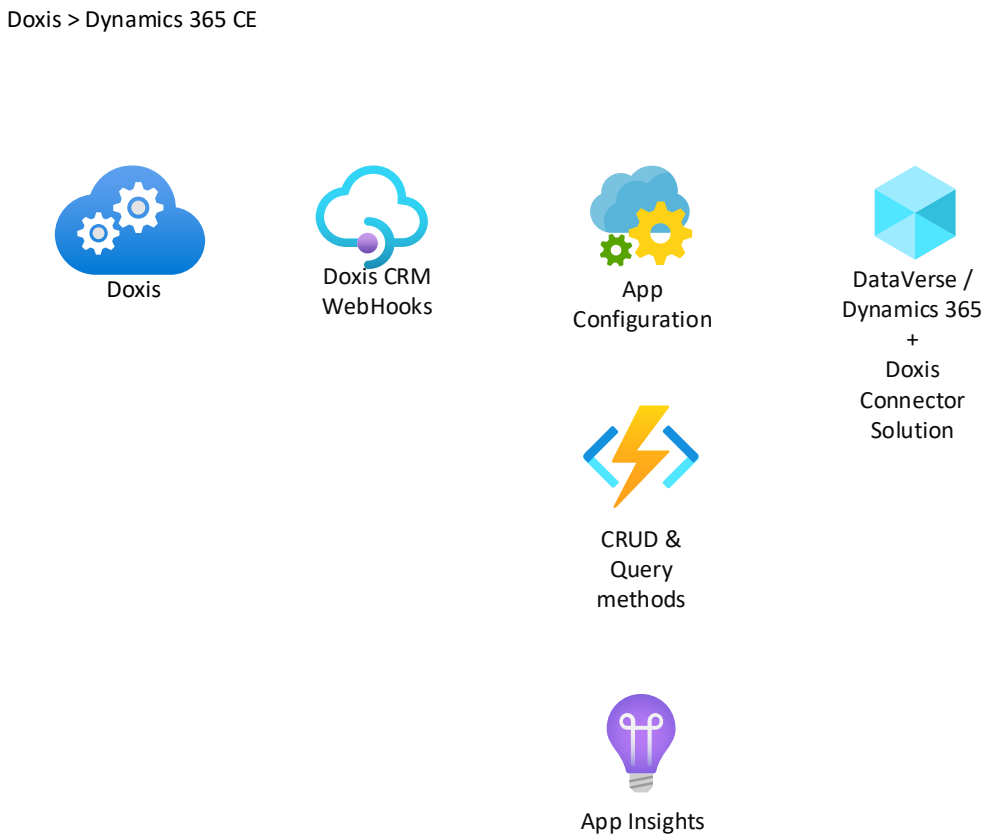
Date	Author	Change Log	Version
14.09.2022	Tom Tajić	Initial version	1.0
27.10.2022	Tom Tajić	Added lookupload descriptions. Doxis parameters descriptions. Changed configuration screen-shots.	1.1
08.12.2022	Tom Tajić	Additional settings descriptions for embedded Doxis web re-source. Added SyncLog entity description.	1.2

Table of Contents

Pictures

1 Solution high-level architecture

Solution architecture supporting the synchronization between Dataverse / CRM and Doxis.



Picture 1 high-level architecture

Architecture components:

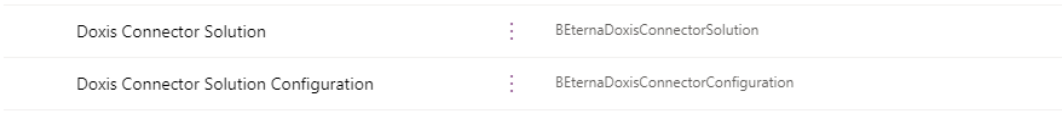
- Doxis is an extremely agile, scalable and secure platform for ECM, BPM and collaboration applications of all kinds. Through APIs and connectors, Doxis provides information, manages cross-system processes and connects to Dataverse / Dynamics CRM and other systems.
- Azure components
    - Azure API Management is a reliable, secure and scalable way to publish, consume and manage API's running on Microsoft Azure platform
        - Doxis CRM WebHooks API
            - Handles events from Dataverse / Dynamics 365 and tranforms them into Doxis API payload
    - Azure App Configuration is an Azure service that allows users to store and manage configuration within the cloud

        - In this key-value storage, for each Doxis CRM WebHooks subscription, a set of configurations is stored
    - Azure App Insights provides Application Performance Monitoring (also known as "APM") features.
- Dataverse / Dynamics 365 organization
    - Installed Doxis Connector Solution

2 Setup instructions

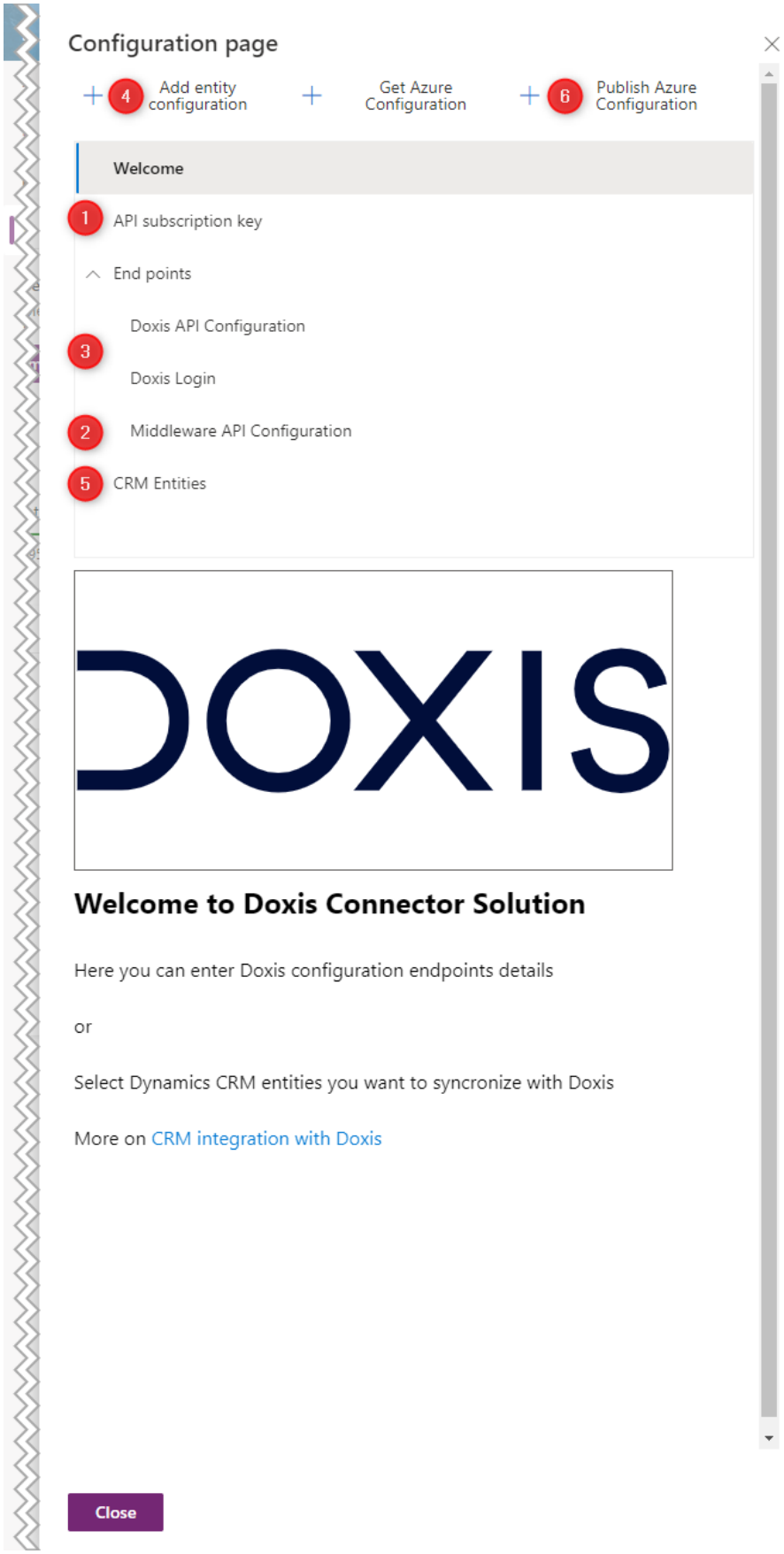
Steps to setup the solution:

1. Customer signs up to Doxis CRM WebHooks API on our API portal
2. After confirming licence obligations you shall receive a welcome email with 2 CRM solutions and instructions
    a. After confirmation you can at any time get the API subscription key on the API portal
3. Doxis Connector Solutions should be downloaded, imported and then published to customers Microsoft Dynamics 365 / Dataverse organization.
    a. Doxis Connector Solution is a managed solution with configuration application logic
    b. Doxis Connector Solution Configuration is an unmanaged solution where your specific configuration is stored and can be manipulated by using the solution's configuration page



Picture 2 CRM Solutions

4. Customer uses Doxis Connector Solution Configuration solution to:
    1) enter API subscription key
    2) set up transformation middleware API specifics
    3) setup Doxis API specifics
    4) add new entity configuration
    5) modify existing entity configuration
    6) publish configuration to Azure


Configuration page
4 Add entity configuration
Get Azure Configuration
6 Publish Azure Configuration
Welcome
1 API subscription key
End points
Doxis API Configuration
3
Doxis Login
2 Middleware API Configuration
5 CRM Entities
DOXIS
Welcome to Doxis Connector Solution
Here you can enter Doxis configuration endpoints details
or
Select Dynamics CRM entities you want to syncronize with Doxis
More on CRM integration with Doxis
Close

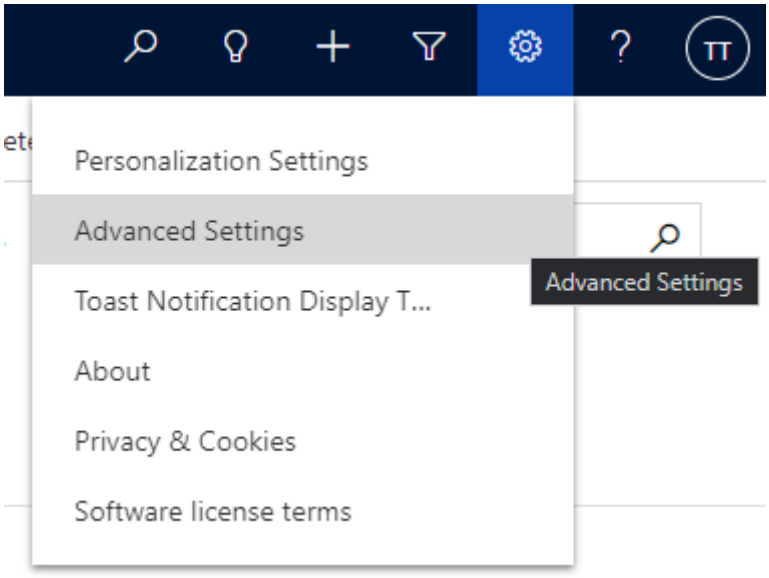
Picture 3 Configuration navigation options

3 Doxis Connector Solution

To use and configure the solution the user needs to have Crm Administrator security role and Dataverse / Dynamics 365 customization knowledge.

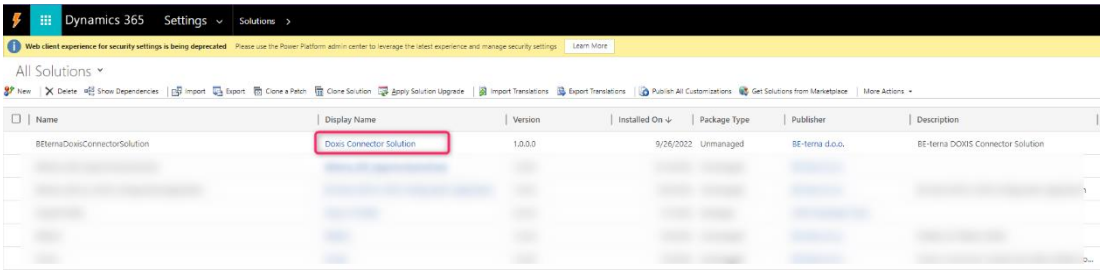
3.1 Open solutions configuration page

- After downloading, import first the:
  - Doxis Connector Solution managed solution
    then
  - Doxis Connector Solution Configuration unmanaged solution
- After importing and publishing the Dynamics CRM solutions you can open Doxis Connector Solution Configuration solution and use the configuration page.
- To open solutions configuration page either:
  - go to https://make.powerapps.com, select correct environment of your Dataverse / CRM organization and got to solutions
  - go to your dataverse / crm organizations settings using link like this: https://<YOUR_ORGANIZATION>.crm<YOUR_REGION>.dynamics.com/main.aspx?settingsonly=true.
    Replace YOUR_ORGANIZATION and YOUR_REGION with data specific to your dataverse / crm organization.
  - or click on settings (gear wheel) / Advanced Settings



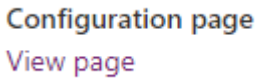
Picture 4 Advanced Settings

- Go to Solutions
- (in advanced settings – old solution UI) Open »Doxis Connector Solution Configuration« and click on »Configuration« in the solutions left-side navigation tree.
-



Picture 5 Doxis Connector Solution

- (in make.powerapps.com)) Open »Doxis Connector Solution Configuration« and click on Overview and then under Configuration page click on View page

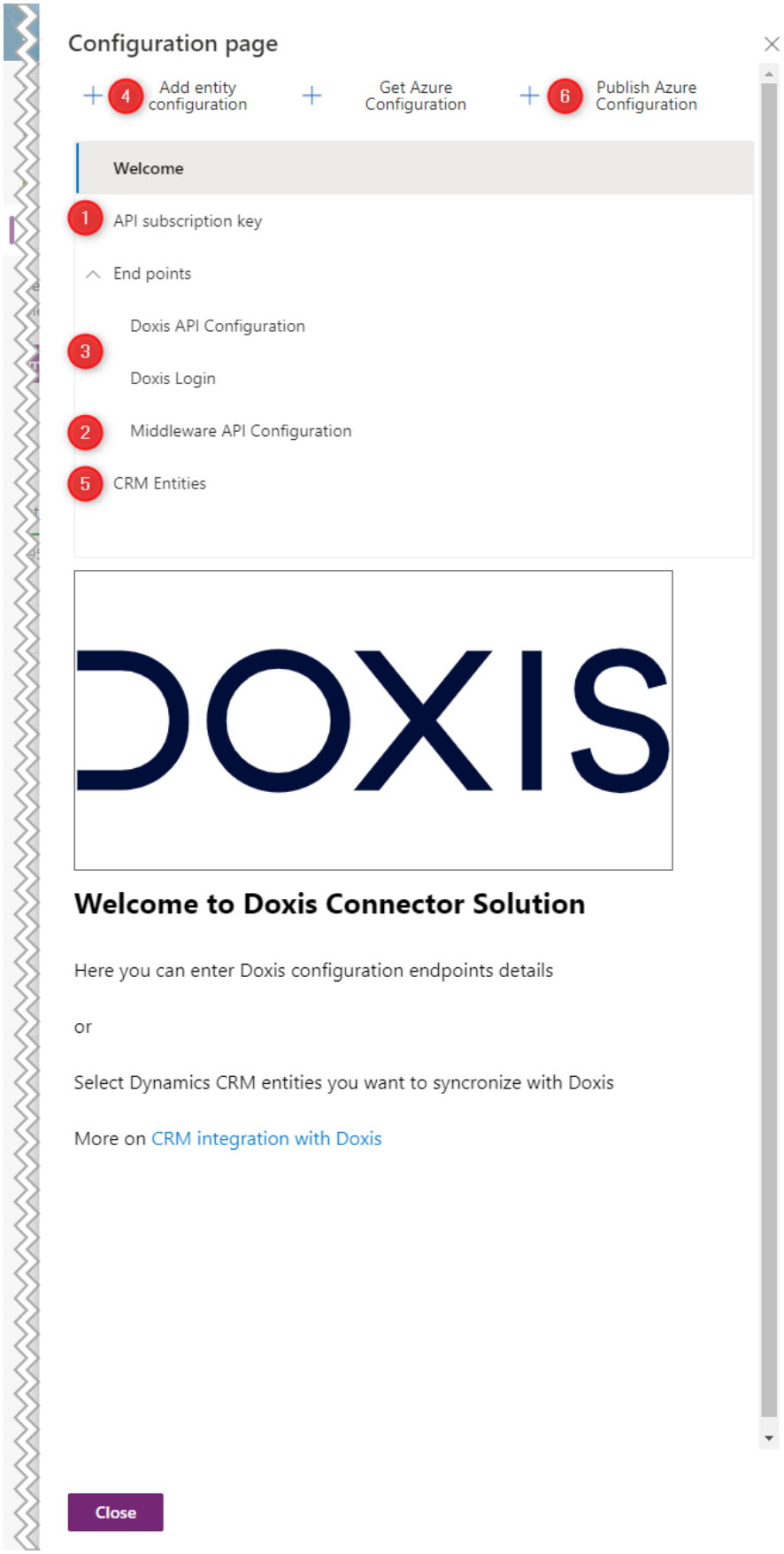


3.2 Doxis Configuration page

This is where all the API and entity mapping and transformation configurations are configured.

Configuration page allows you to:

1) enter API subscription key
2) set up transformation middleware API specifics
3) setup Doxis API specifics
4) add entity configuration
5) modify existing entity configuration
6) publish configuration to Azure

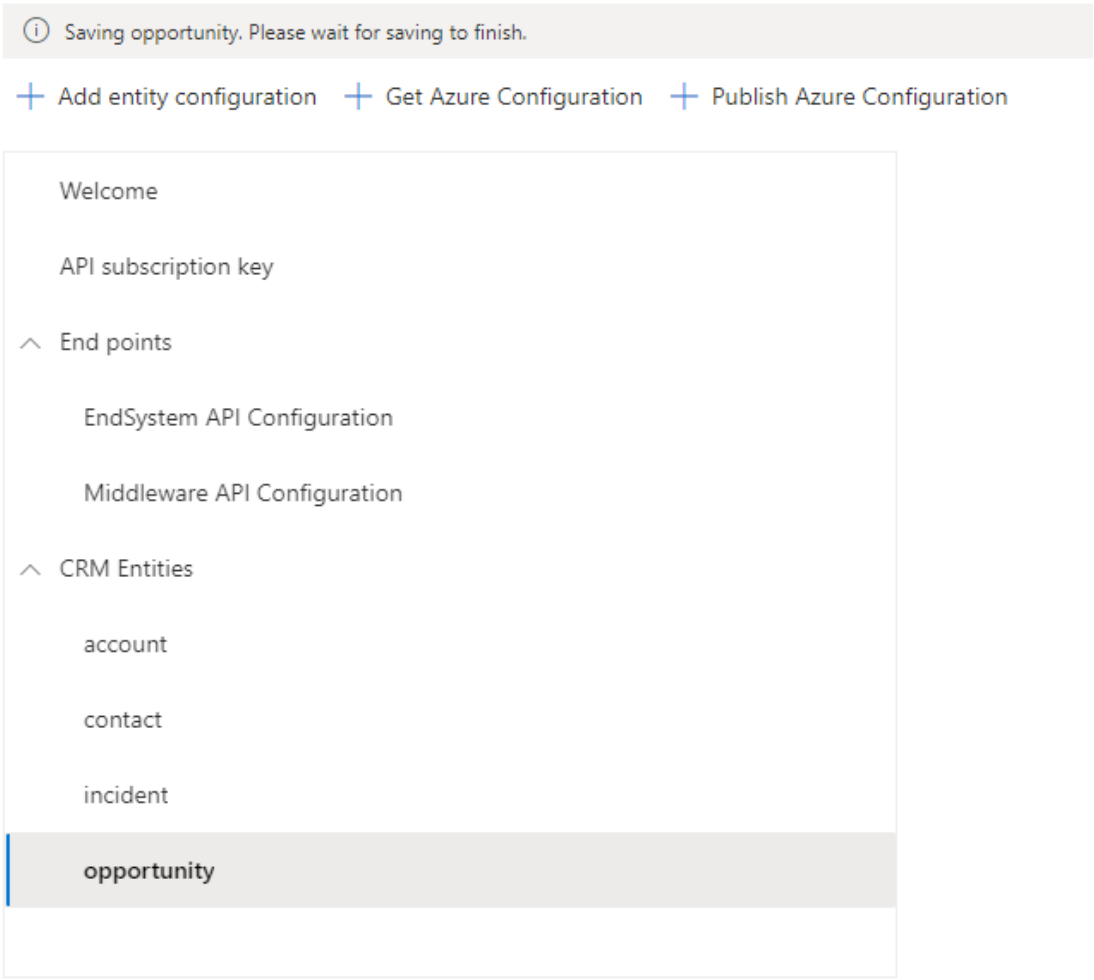

Configuration page
4
Add entity configuration
Get Azure Configuration
6
Publish Azure Configuration
Welcome
1
API subscription key
End points
Doxis API Configuration
3
Doxis Login
2
Middleware API Configuration
5
CRM Entities
DOXIS
Welcome to Doxis Connector Solution
Here you can enter Doxis configuration endpoints details
or
Select Dynamics CRM entities you want to syncronize with Doxis
More on CRM integration with Doxis
Close

Picture 6 Configuration navigation options

3.2.1 Saving the configuration (important)

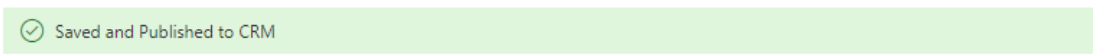
When you save the changes (»Save« button), the result of configuration (json) is stored (saved and published) as a webresource named »bt_/resx/webhookconfiguration.resx« in Dataverse / CRM.

When you click »Save« a gray notification is displayed, that informs you to wait for changes to be saved. This is important, because if you navigate away the changes might not save and you might loose your changes.



Picture 7 Saving changes - Need to wait for saving to finish

After successfully saved a green notification is displayed.



If you want to use the configuration to be used in transformations you need to click »Publish Azure Configuration« button.

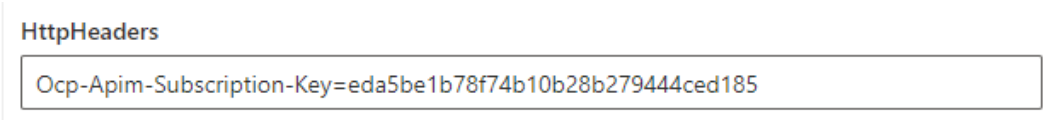
3.2.2 Step1: API subscription key

Enter API subscription key that you have received when subscribing to the Doxis CRM WebHooks API through API Portal.

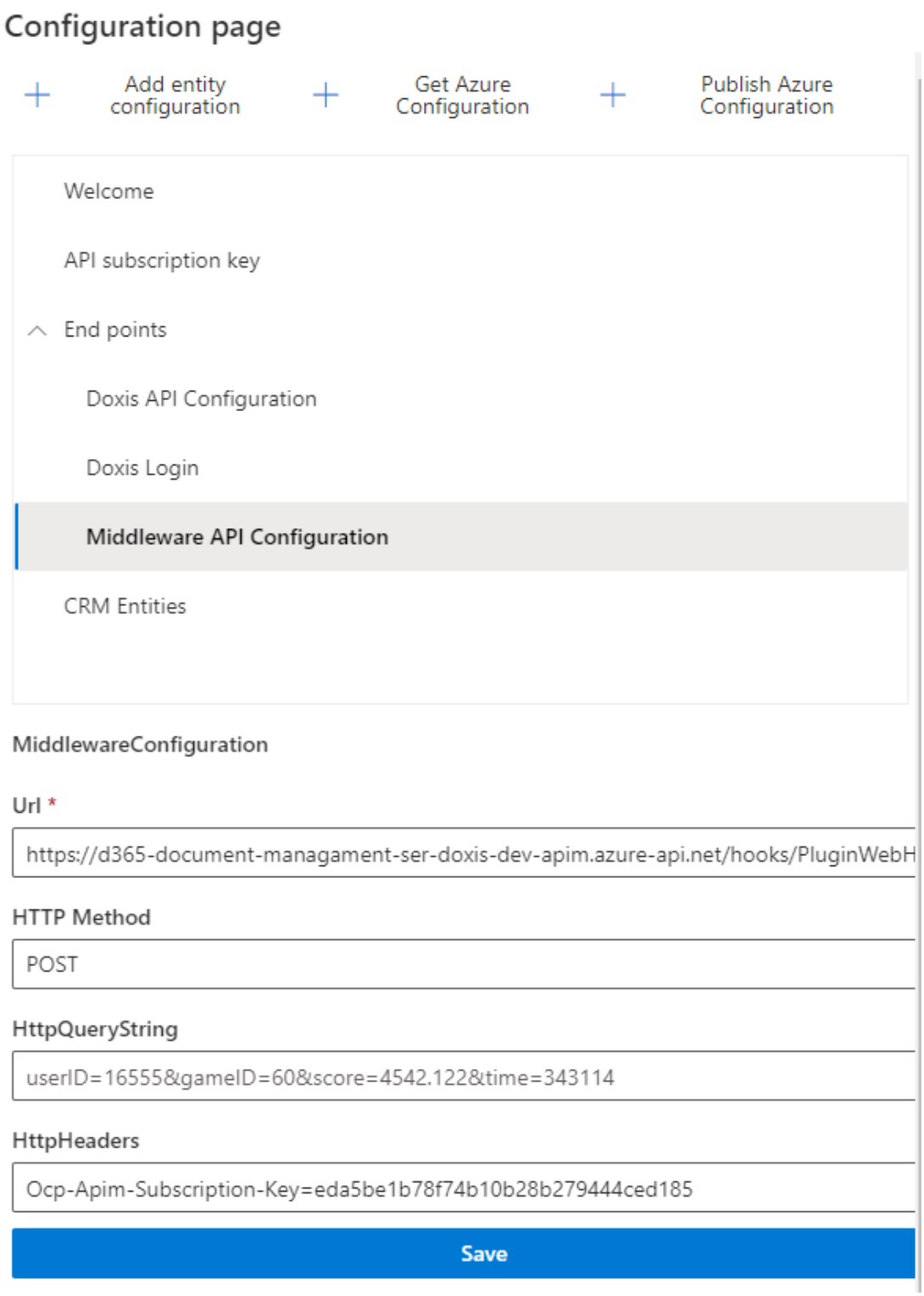
After that save the configuration, wait for the save to be completed and after that publish azure configuration.

3.2.3 Step2: Modify Middleware API Configuration

Replace existing subscription key in HttpHeaders with yours.



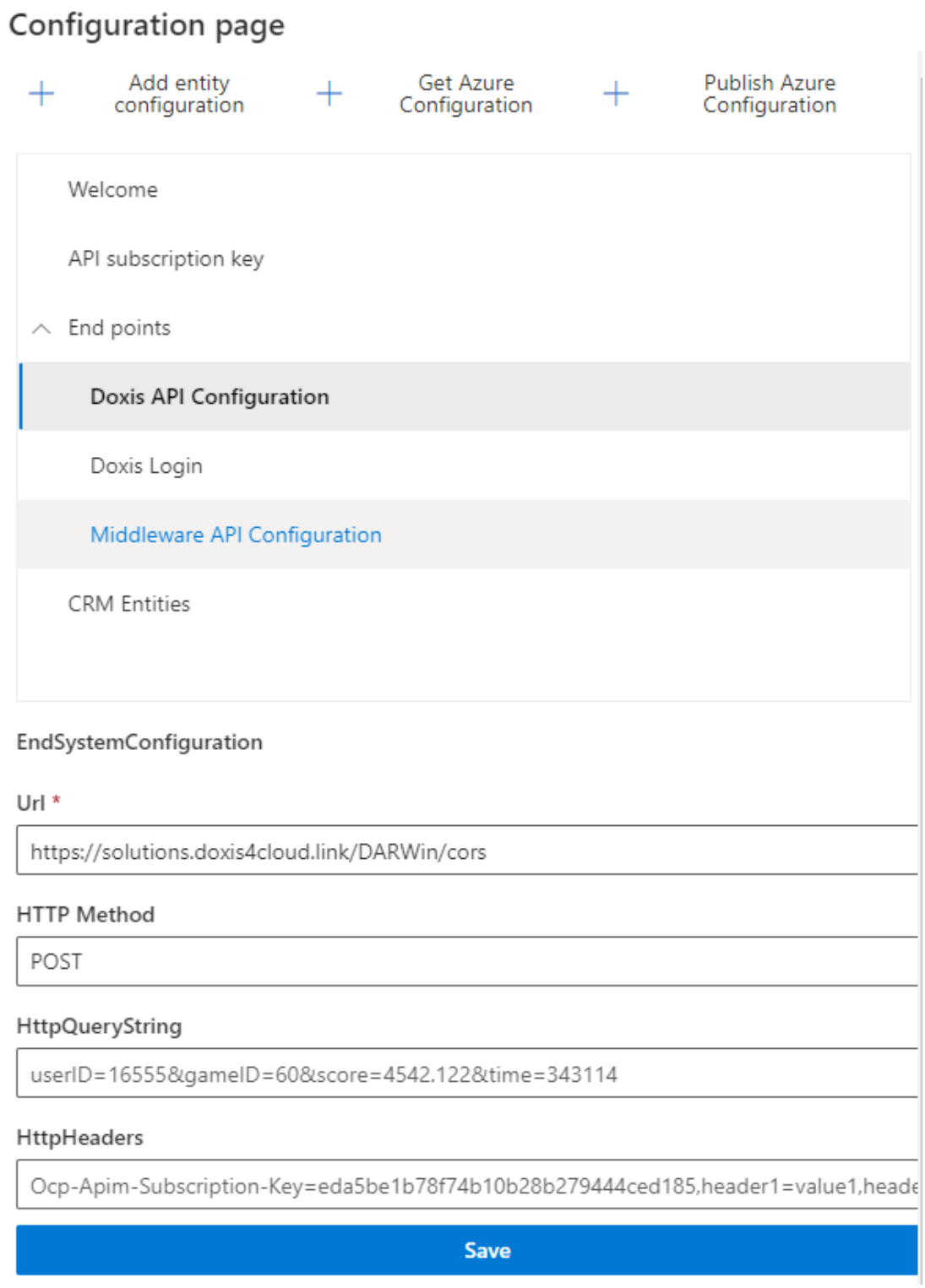
Other information should only be changed if instructed by the BE-terna support team (if new API version occurs, azure reconfiguration was done, …)



Picture 8 Middleware Configuration

3.2.4 Step3: Doxis API Configuration

Here you configure your Doxis endpoint.

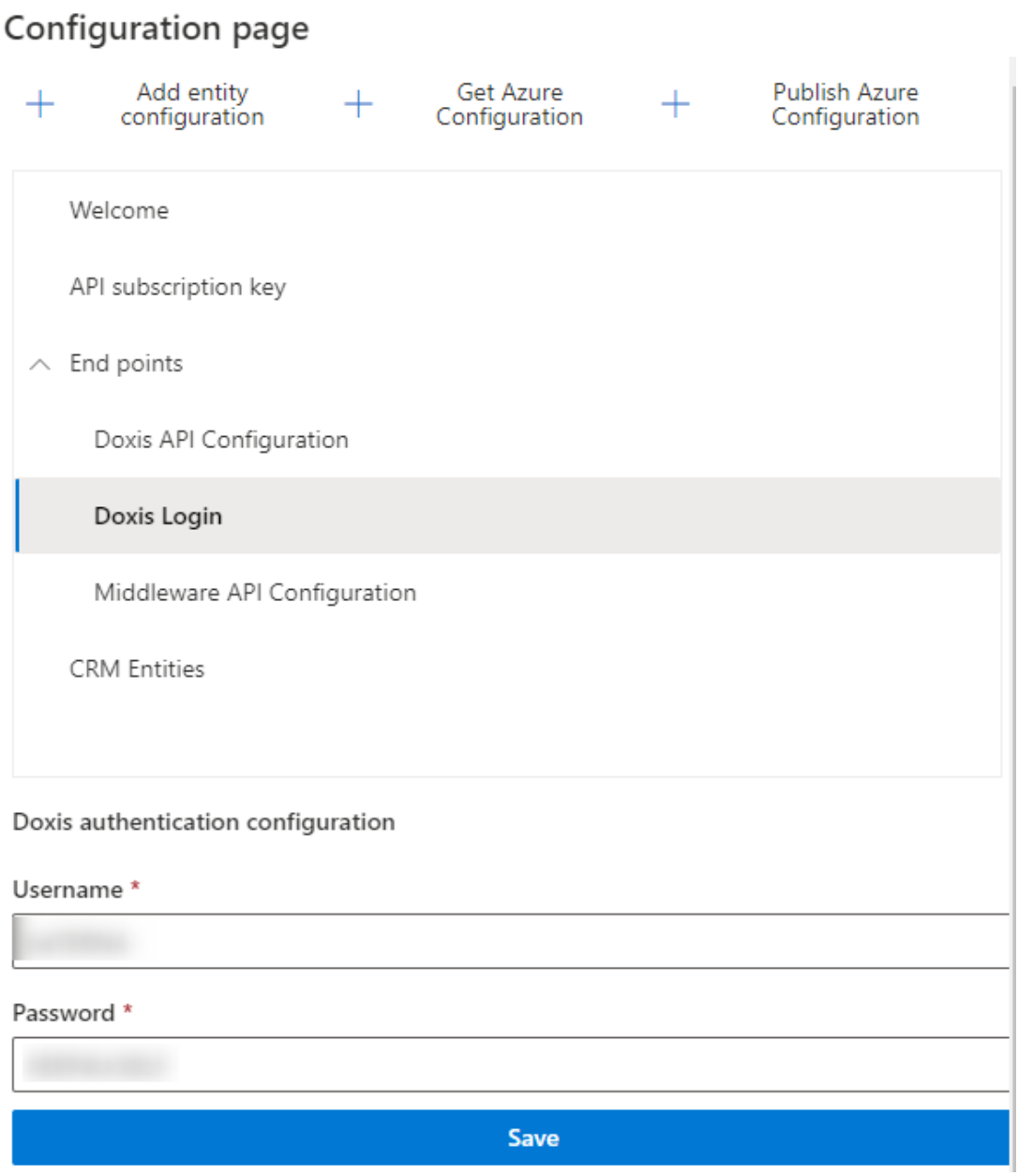


Information needed:

1. Doxis API endpoint url
2. HTTP method
3. Http Query String
4. Http Headers

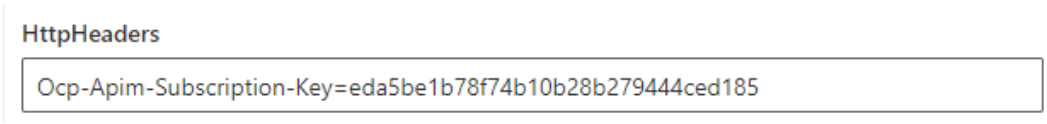
Doxis login information

5. Username (Doxis user used in synchronization)
6. Password



Picture 9 Enter Doxis API username and password

Replace existing subscription key in HttpHeaders with yours.



3.2.5 Step4&5: Add / Modify entity configuration

Form to add or modify entity configuration is the same.

CRM Entity *

Opportunity

DOXIS FolderClass *

OPPWorkspace

Enter DOXIS folder class

DOXIS RESTCube URL *

/CubeFolder/OPPWorkspace

Enter DOXIS RESTCube URL (this is an URL in API Management that wraps RESTCube URL)

DOXIS Open URL *

https://solutions.doxis4cloud.link/webcube/?server=BSOL&system=dx4bsol&action=showse...

Enter DOXIS URL used to open or embed DOXIS folder from CRM

Liquid templates introduction

Liquid template *

```
{
  "UniqueID": "{{_entityId}}",
  "ObjectType": "OP00",
  "ObjectName": "{{name}}",
  "Date": "{{createdon|date:"yyyy-MM-dd"}}",
  "ObjectAnnotation": "{{description}}",
  "ObjectState": "OP00",
  "DateStart": "{{createdon|date:"yyyy-MM-dd"}}",
  "DateEnd": "{{actualclosedate|date:"yyyy-MM-dd"}}",
  "SystemModstamp": "{{modifiedon|date:"yyyy-MM-ddTHH:mm:ss.fffZ"}}",
  "PartnerLink_Ext": "{{parentaccountid}}"
}
```

Enter liquid template that is used to map existing CRM attributes to DOXIS folder elements

Edit Attributes (8 attributes mapped)

Save

Picture 10 Add / Modify entity configuration

All Doxis fields should be obtained from SER and Doxis swagger page associated to your Doxis setup.

DOXIS Folder Class
DOXIS Folder Class is an unique Doxis folder descriptor.

DOXIS RESTCube URL
This is a partial RESTCube path to specific Folder Class (example: /CubeFolder/OPPWorkspace). You can easily find this paths if you open RESTCube Swagger – please see Picture 11 Doxis Swagger – Opportunity.

RESTCube API Swagger is typicaly found on url like this:
https://<your doxis solution server>/DARWin/swagger-ui/?url=/DARWin/pool/CubeSwagger/json

Doxis Open URL should be configured like this:

https:/<WebCube application url>/webcube/?server=Profilename&system=Organisation&action=showsearch&id=SearchID&UniqueID=CHANGEME&nodeid=NodeID&openone=1&dateFrom=0&embed=1&disableclosecon-firm=1&home=1&view=1&splash=0&layout=defaultclassic&topbar=0&sam-llogin=true&reusesession=1&

Adjust the following variables:

- WebCube application url

example: solutions.doxis4cloud.link

- Profilename

Name of the server profile defined in the Doxis4 webCube administration console.

- Organisation

Name of an organization of that server.

- SearchID

GUID of the search class

- NodeID

GUID of a node. When a record is displayed, the navigation tree is expanded and the corresponding node is opened.

- UniqueID

needs to be the name of the Control in the referenced Search dialog, where the unique ID of the CRM entity has to be entered.

By default, also the following URL parameters are used:

- openone=1

Opens a single document directly if it is the only object found in a search.

- dateFrom=0

Sets a search period with an open start.

- embed=1

Prevents allocation of a new session ID when the logon status changes.

- disablecloseconfirm=1

Suppresses the display of a prompt when Doxis4 webCube is quit.

- home=1

Suppresses the display of a link to the Doxis4 webCube home page.

- view=1

Suppresses the display of tabs. Each information object subsequently displayed will be opened in a separate window using the same session.

- splash=0

Suppresses the display of a splash screen.

- layout=defaultclassic

Displays classic-style controls.

- topbar=0

Hides the Doxis4 webCube bar.

- samllogin=true

Forces SAML authentication.

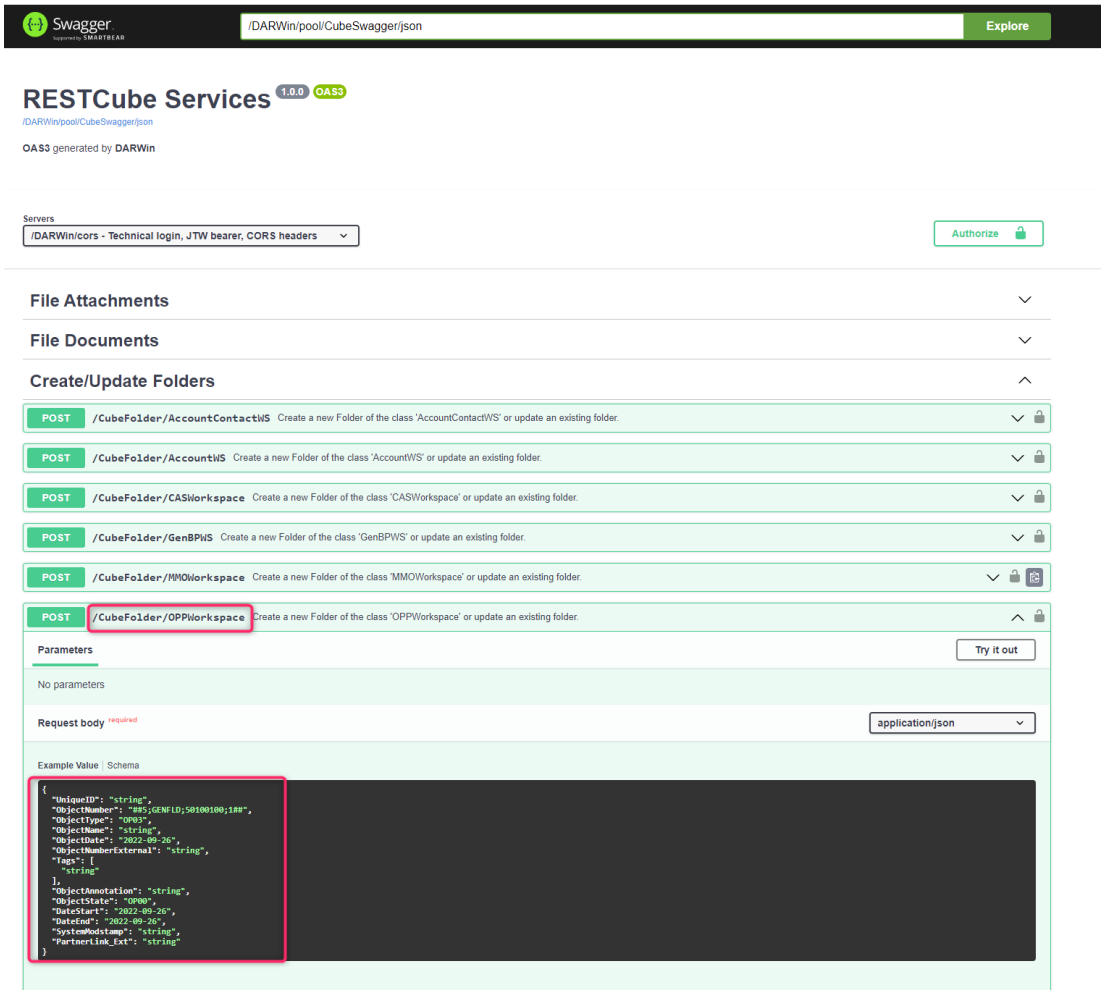
- reusesession=1

Configures use of an active session.

Here you also define the liquid template, that is used to transform Dataverse / CRM selected entity attributes to final JSON Doxis payload representation.

First you should obtain the JSON from Doxis swagger for the entity you are configuring and copy / paste it in the liquid template field.

For opportunity it looks like this



Picture 11 Doxis Swagger – Opportunity

After you have the Doxis JSON in the liquid template, start adding required fields to the entity configuration (Edit attributes button).

You can then edit the liquid template to map the attributes you have added by using the liquid template syntax.

3.2.6 Edit attributes

When editing entity configuration, you have an option to selected entity attributes that will be sent to Doxis upon create / update or on-demand sync events.

The form looks like this, and allows you to:

1. Select CRM Attribute for current entity and »Add« it
2. Delete CRM Attribute from configuration

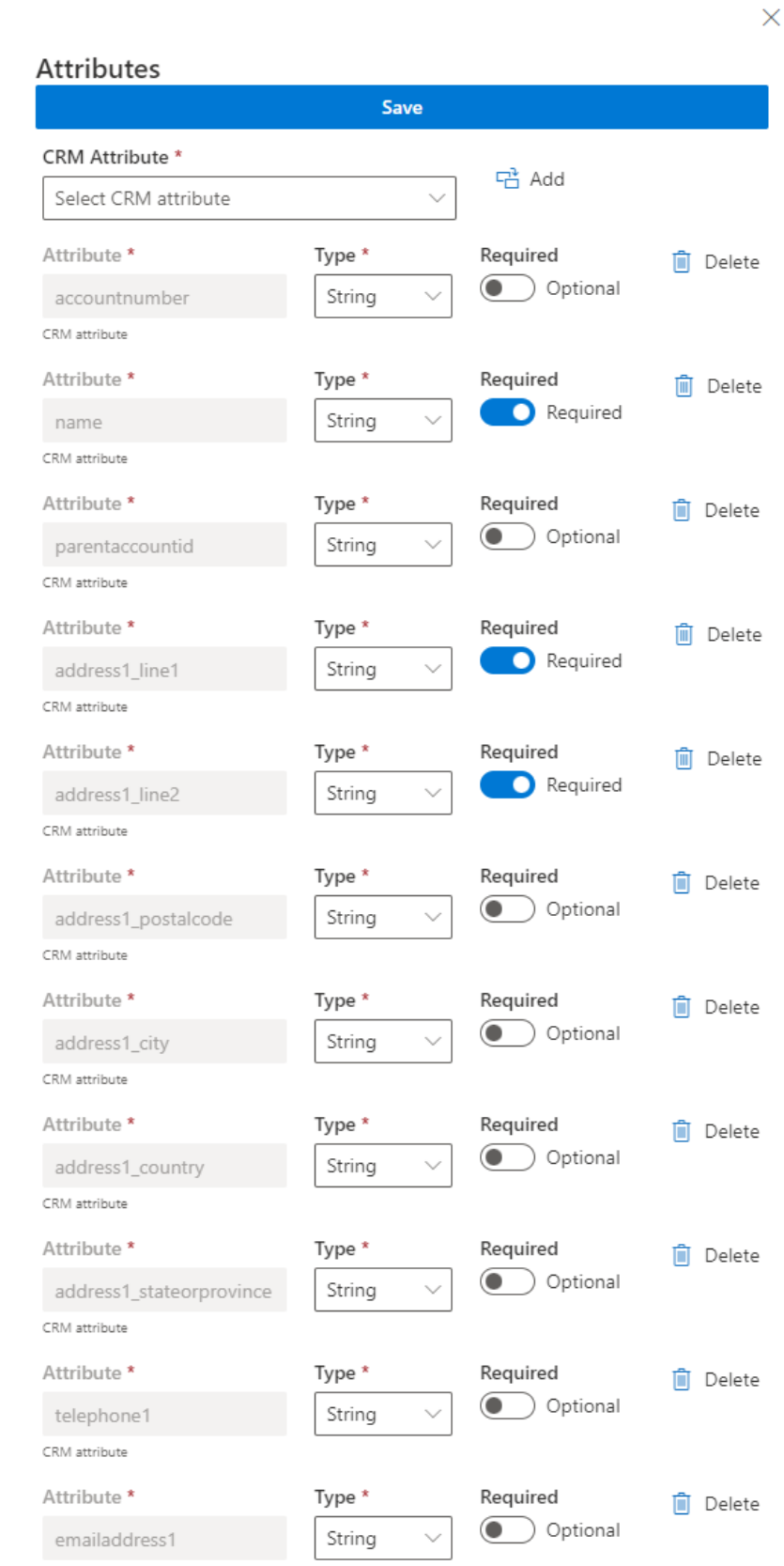
3. CRM attribute name should be used in liquid template to map the value to destination attribute. If you are not using the attribute, you should remove it from configuration (although there is nothing wrong if you leave it)
4. Specify type (described below)
    - String (for certain CRM field types (optionset field, lookup field) this means it will transfer fields text value rather than actual value)
    - int (should be used only for optionset field types)
    - lookupload
        - should be used only for EntityReference field types – if you happen to use it on non-entityreference type it is simply outputting the value as string).
        - lookupload tries to "load" the configuration attributes of entity specified in selected attribute of type entityreference and allows you to read its attribute. Example: if you specify in your liquid template:

"Contact": "{{primarycontactid.fullname}}"

and primarycontactid is a entityreference to contact entity and you have provided the configuration for contact entity containing the attribute fullname, then fullname from that contact will be passed as value.

5. Required / Optional
    - Required: value is always sent, regardless if it has changed or not
    - Optional: value is only sent when changed (it is ommited if not changed)

Configuration page

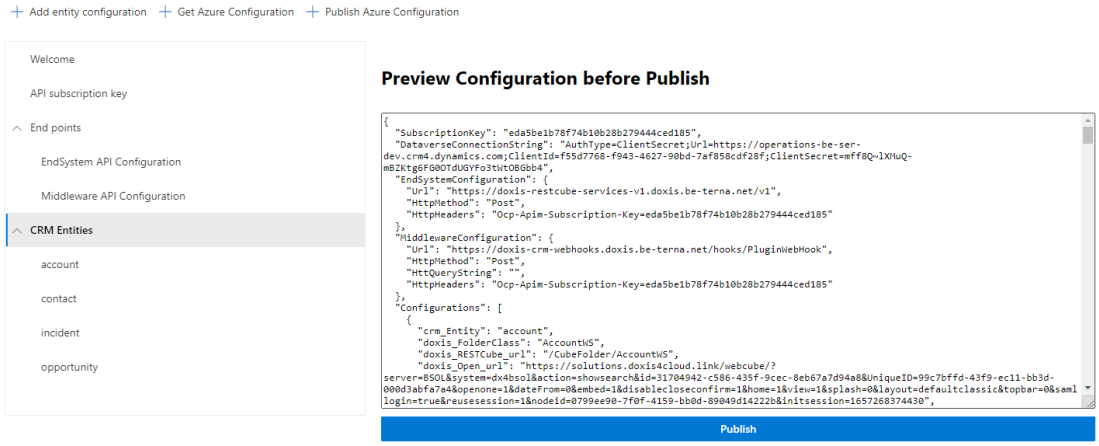


Picture 12 Example of attributes configuration for account

3.2.7 Step 6: Publish Azure Configuration

User needs to publish the configuration to Azure, so that middleware can transform the data to Doxis payload and send it to Doxis.

User selects »Publish Azure Configuration« button and is presented with »Preview Configuration« screen.

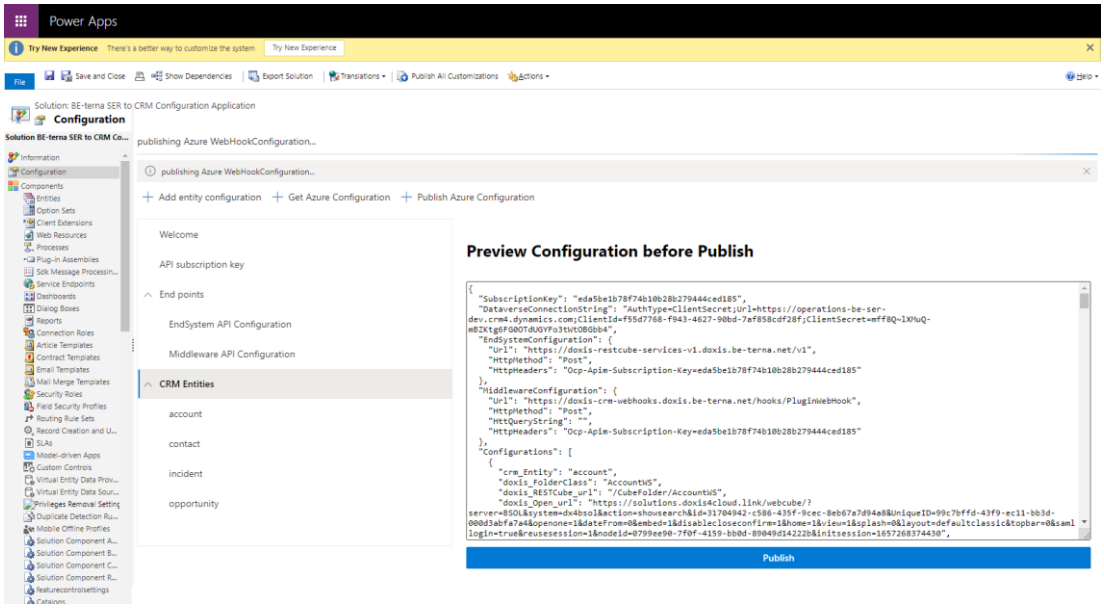


Picture 13 Preview configuration screen

User can inspect the json configuration to be sent to Azure.

When user is sure that configuration is correct, he selects the »Publish« button.

Gray notification »publishing Azure WebHookConfiguration« is displayed.



Picture 14 Publishing Azure Configuration notification

After publishing user is presented with »Published to Azure« notification

Picture 15 Published to Azure notification

3.3 Basic Liquid Template Syntax

This is an example where basic liquid template syntax will be explained.

Bolded parts of liquid template are further explained below the example.

Example:

```
{
  "UniqueID": "{{_entityId}}",
  "PartnerNumber": "{{accountnumber}}",
  "PartnerName": "{{name}}",
  "AccountID": "{{_entityId}}",
  "ParentID": "{{parentaccountid}}",
  "Street": "{{address1_line1|append:" "}}{{address1_line2}}",
  "PostCode": "{{address1_postalcode}}",
  "City": "{{address1_city}}",
  "State_Region": "{{address1_stateorprovince}}",
  "Country": "{{address1_country}}",
  "Phone": "{{telephone1}}",
```

```
  "EMail": "{{emailaddress1}}",
  "Fax": "{{fax}}",
  "URL": "{{websiteurl}}",
  "Type": "{{customertypecode}}",
  "Industry": "{{industrycode}}",
  "Description": "{{description}}",
  {% if statecode == 0 %}
  "Deactivated": false,
  {% else %}
  "Deactivated": true,
  {% endif %}
  "SystemModstamp": "{{modifiedon|date:"yyyy-MM-ddTHH:mm:ss.fffZ"}}"
}
```

3.3.1 Mapping entity unique identifier

_entityId is a special attribute name that is the only field, you do not need to manually add to the entity configuration.

It is mapped like this:
"UniqueID": "{{_entityId}}",

_entityid is actually a GUID (example: 939082c1-354d-4e44-98c9-0acabdcaaed1)

3.3.2 Concatination of multiple fields into single Doxis attribute

Concatination example:
"Street": "{{address1_line1|append:" "}}{{address1_line2}}",

Liquid template append statement appends empty space only if address1_line1 is not empty.

You could also do this if you would like to concatanate values regardless of the value existance:
"Street": "{{address1_line1}}{{address1_line2}}",

3.3.3 Conditional mapping

For certain mappings you can use also if else statements:

```
  {% if statecode == 0 %}
  "Deactivated": false,
  {% else %}
  "Deactivated": true,
```

{% endif %}

3.3.4 Date-time formatting

Date-time formatting should always be used, so that values are correctly mapped into end systems date-time formats.

Example:

"SystemModstamp": "{{modifiedon|date:"yyyy-MM-ddTHH:mm:ss.fffZ"}}"

3.3.5 Referenced entity attribute (lookupload)

To use attribute of the referenced entity, you should map it like this:

"PostCode": "string",
"City": "string",
"Contact": "{{primarycontactid.fullname}}",
"State_Region": "string",
"Country": "st",

- Specify the name of the attribute of type entityreference (primarycontactid in our example)
- Followed by dot "."
- Followed by name of the attribute on the entity which primarycontactid from our example is pointing to. To use the values for the referenced entity, you have to add the referenced entity to the configuration and add all attributes you might be referring to. You do not need to register the referenced entity to PluginWebHook if you do not intend to send it to Doxis, and you can put anything in the Doxis url parameters.

Lookupload attribute loads related entity like this (example where main entity account has lookupload on ownerid attribute, which in turn loads configured systemuser attributes as part of the message):

```
{
        "_entityLogicalName": "account",
        "_entityId": "9ca3a32e-41f9-ec11-bb3d-000d3abfa7a4",
        "name": "Starfleet Command",
        "address1_line1": "Starfleet Avenue 11",
        "address1_line2": "Command headquarters  26",
        "fax": "+01 65478999975",
```

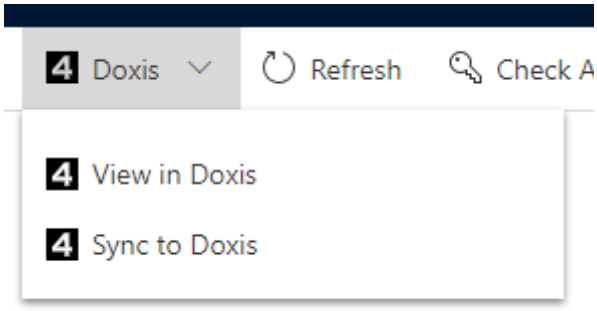
"description": "The Federation Alliance was the accord between the United Federation of Planets, the Klingon Empire, and the Romulan Star Empire to combat the Dominion during the Dominion War. The first offensive undertaken by all the members of the Alliance was the First Battle of Chin'toka in 2374.",
"statecode": 0,
"modifiedon": "10/27/2022 2:37:09 PM",
"ownerid":
{
"_entityLogicalName": "systemuser",
"_entityId": "416acff1-9227-ed11-9db1-000d3aab135d",
"fullname": "CRM.test.user CRM.test.user"
}
}

3.4 Adding ribbon buttons

Ribbon customization should be done by Dataverse / Dynamics CRM developer, so this instructions are presuming you are well versed in ribbon customizations.

There are two ribbon actions that you can decide to copy:

6. View in Doxis (opens related Doxis record in a new window)
7. Sync to Doxis (syncs current CRM record to related Doxis record)



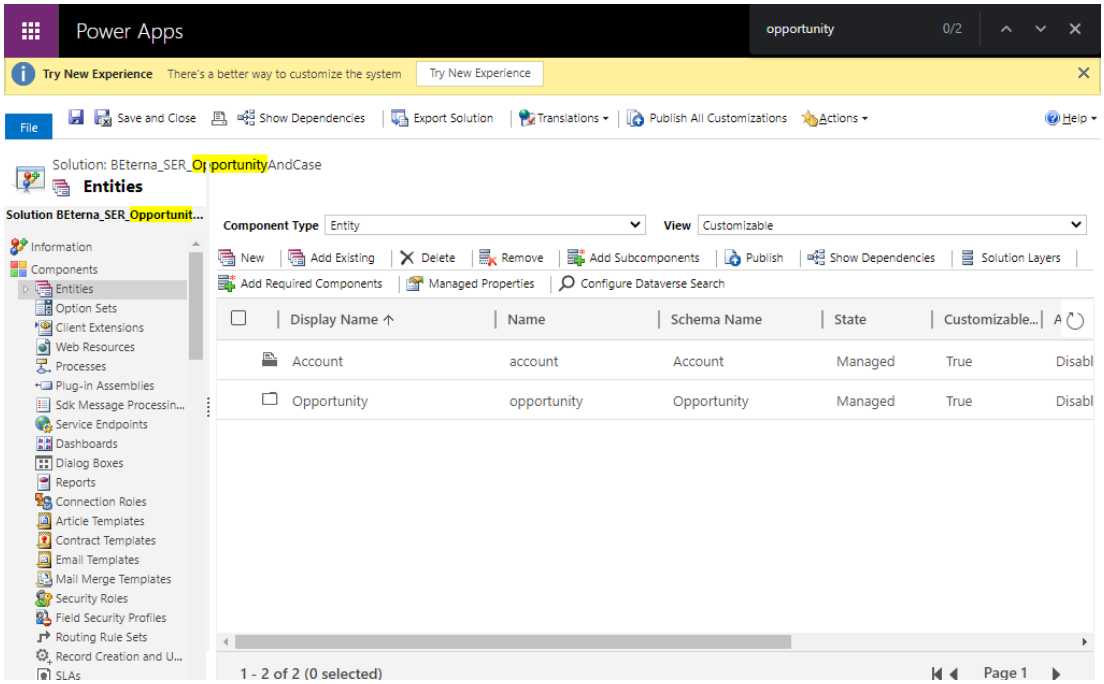
Picture 16 Ribbon actions

Steps to add a ribbon button:

8. Prepare a CRM ribbon solution
9. Open XrmToolbox
10. Copy buttons from »account« entity to your entity

3.4.1 Step1: CRM ribbon solution

First prepare a new CRM solution and put just an account and the entity of your choice (in our case opportunity) in the solution. Then save and publish the solution.

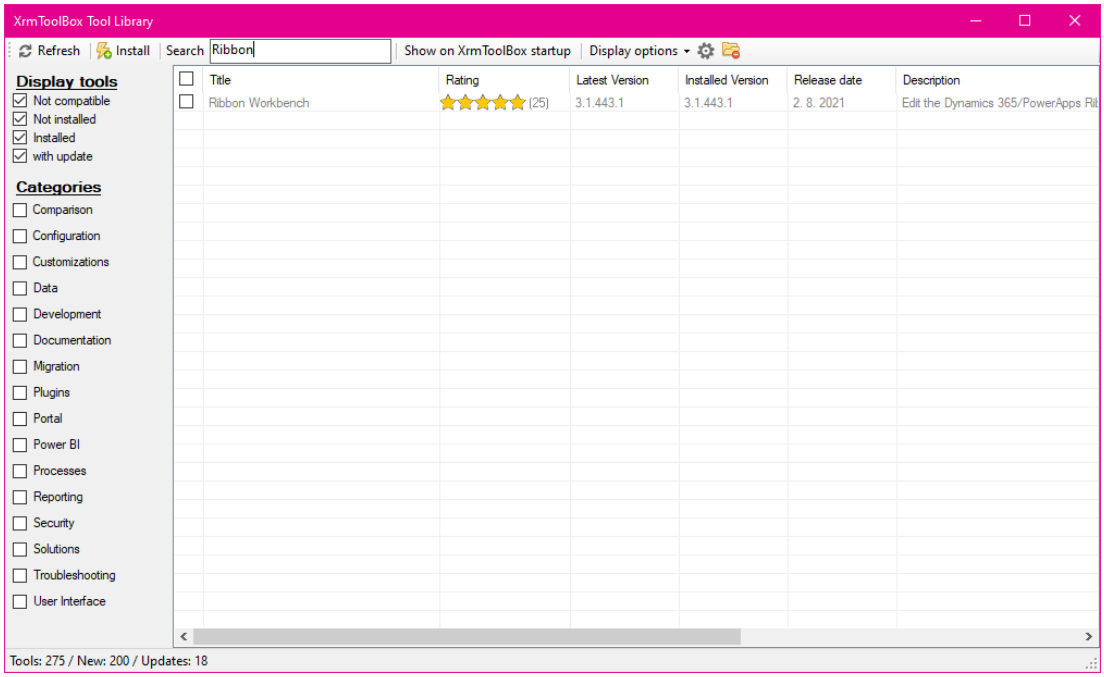


Picture 17 Solution for ribbon customization

3.4.2 Step2: Open XrmToolbox Ribbon WorkBench

On site https://www.xrmtoolbox.com/ you can find and download the latest version of XrmToolbox.

Install it and open a plugin named »Ribbon Workbench« by Scott Durow.
Plugin need to be installed first by using XrmToolbox Tool Library (from Configuration menu).



Picture 18 Search for Ribbon Workbench and install

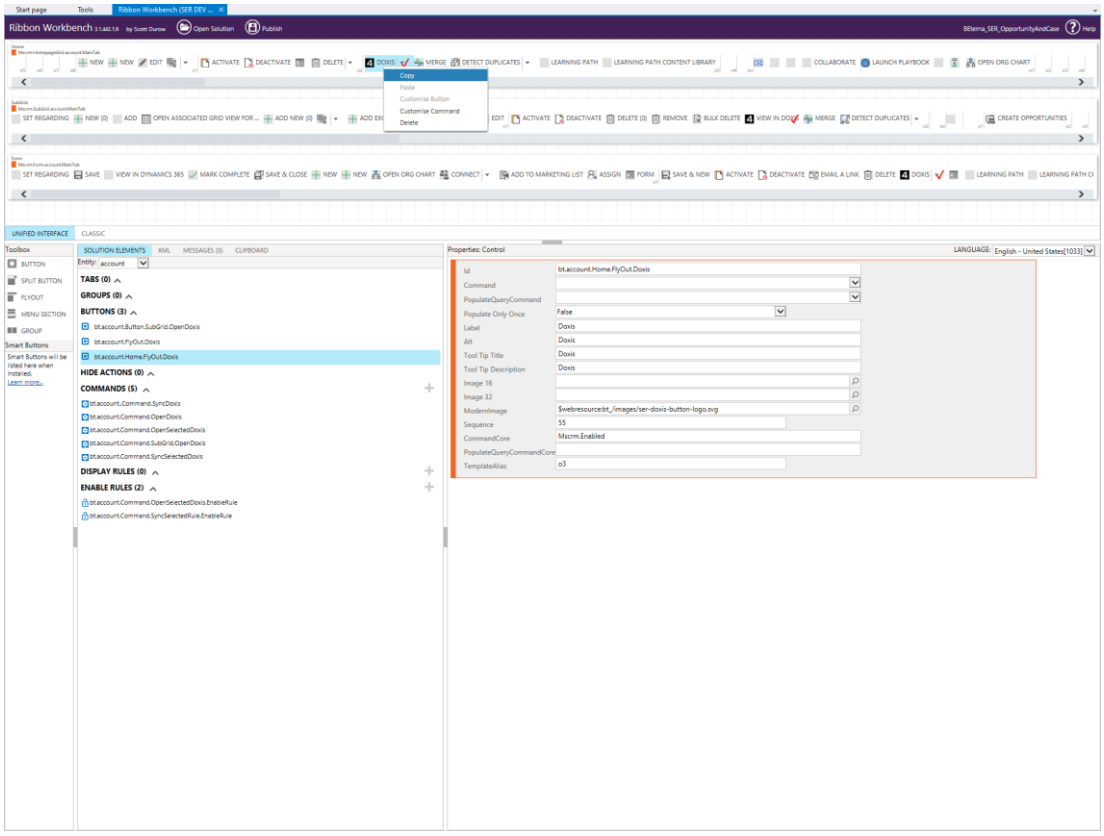
Picture 19 Ribbon Workbench - XrmToolbox plugin

Open the Ribbon Workbench connecting to your Dataverse / CRM organization and select the solution you have prepared for ribbon customization.

3.4.3 Step3: Copy button from "account" entity

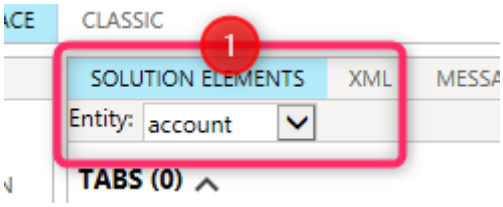
In »Ribbon Workbench« select »account« and copy buttons one by one according to your needs.

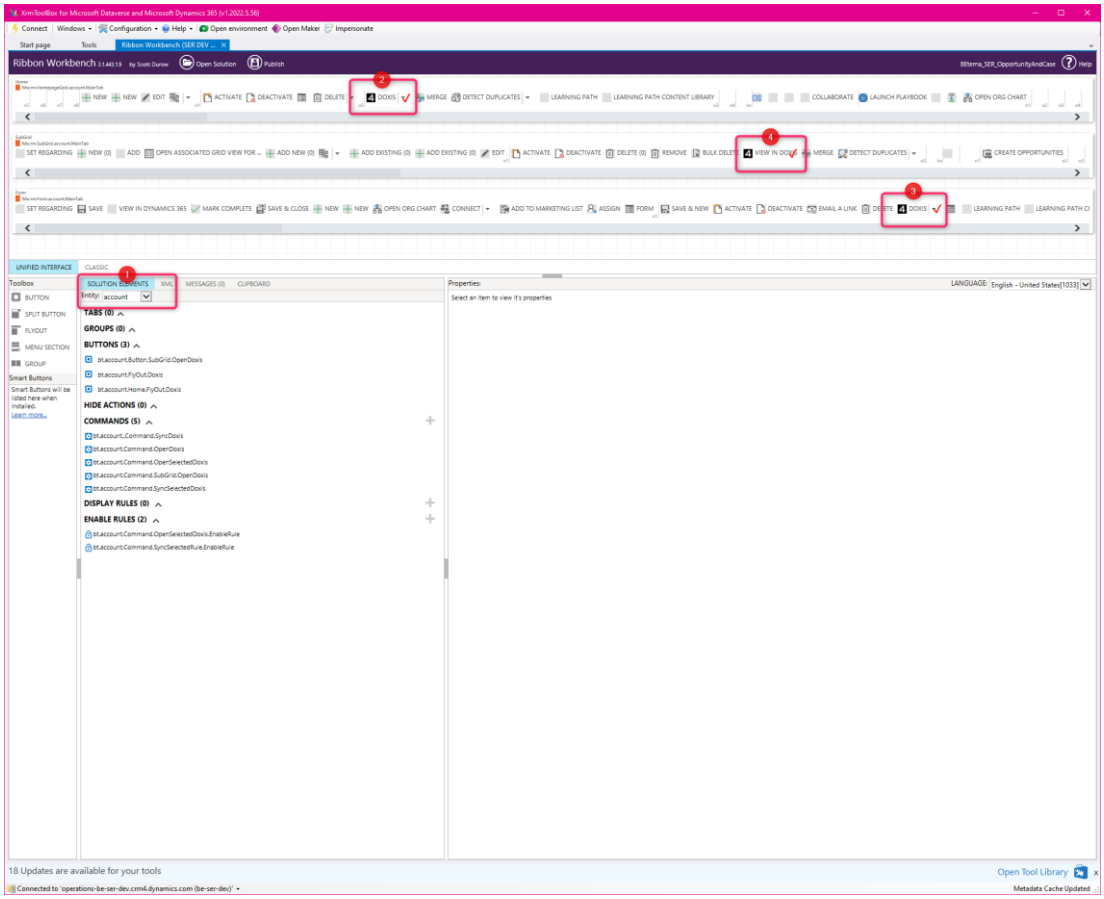
Each menu option need to be copied separately (not the entire ribbon dropdown menu is copied).



Picture 20 Copy button from account

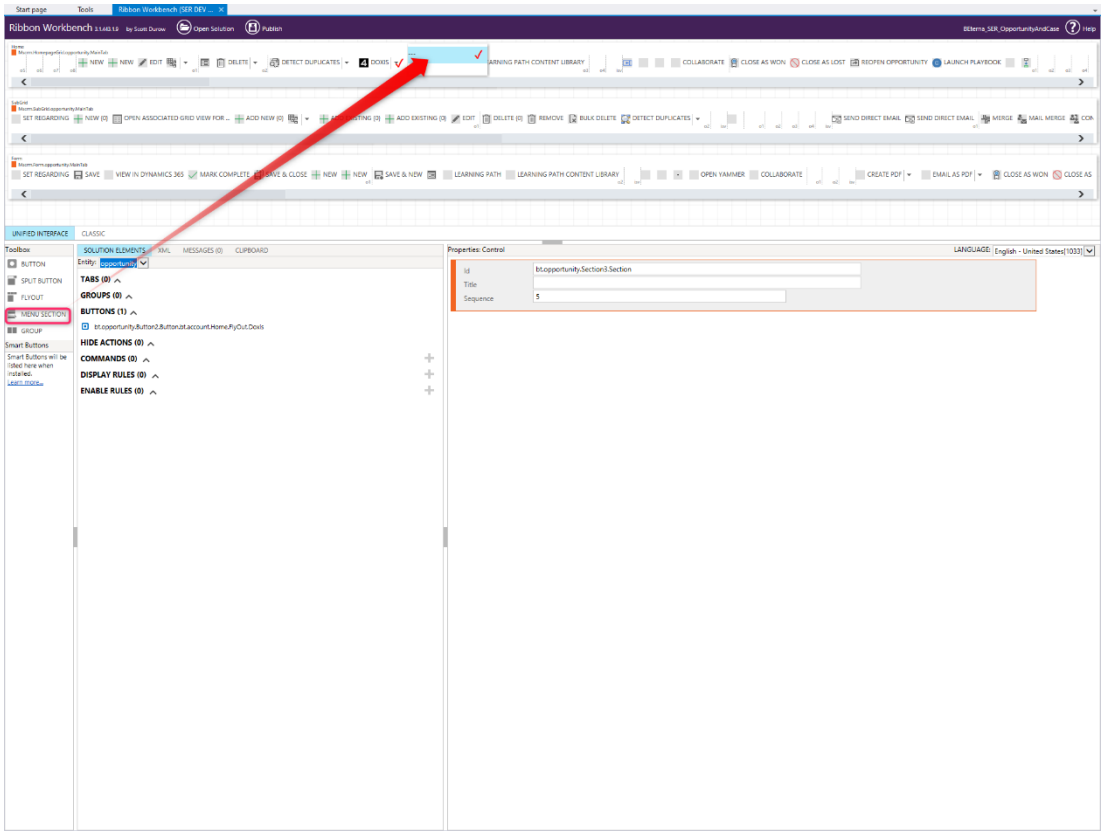
So you will be switching between »account« and »your entity« and copy pasting each ribbon element (button, menu options, …).





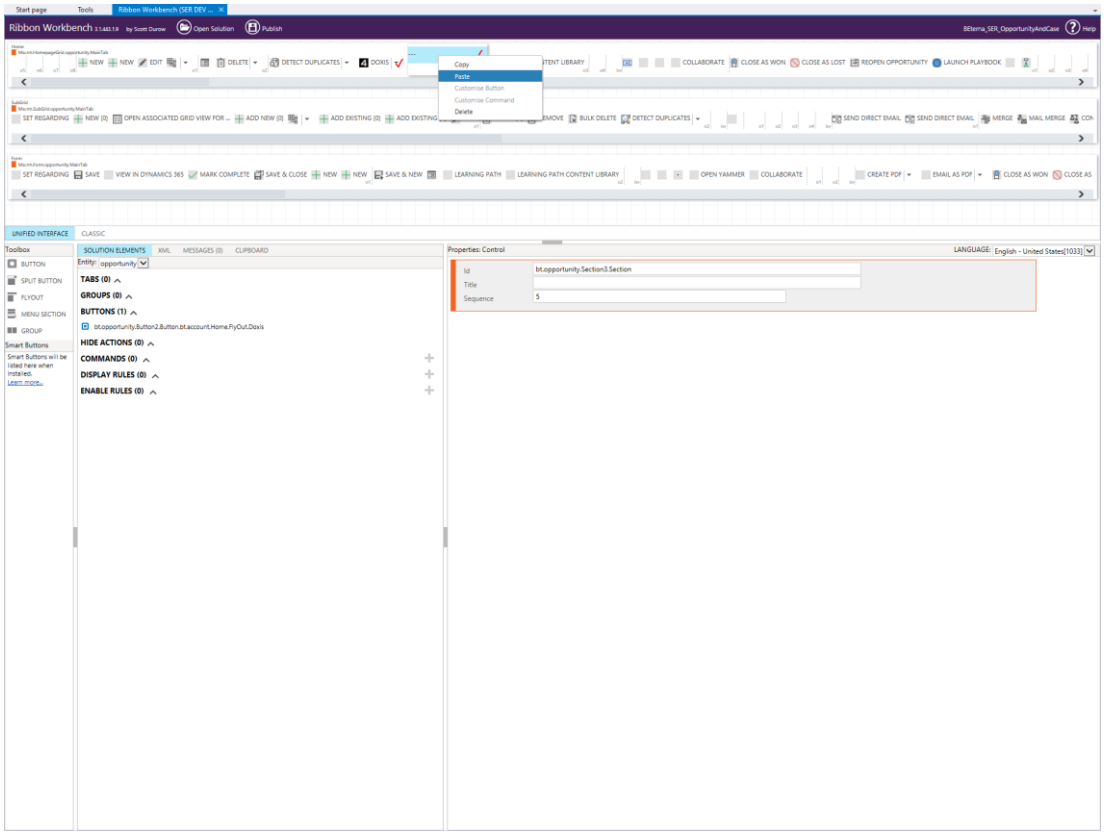
Picture 21 Select account and copy buttons 2, 3, 4 one by one according to your needs

After copying the button from account to your entity, you need to drag & drop the MENU SECTION onto it. This is the placeholder for the menu options that you will copy from account.



Picture 22 Drag & drop the menu section on the button you copied.

After copying the button it is time to copy menu elements from account. You paste the menu elements to destination by clicking on first menu section option where you would like it to be and right clicking it to get the "Paste" option…see below.



Picture 23 Pasting elements

(optional) After copying you should rename your buttons and related commands to make sense as their names still relate to account, but should be named after your entity.

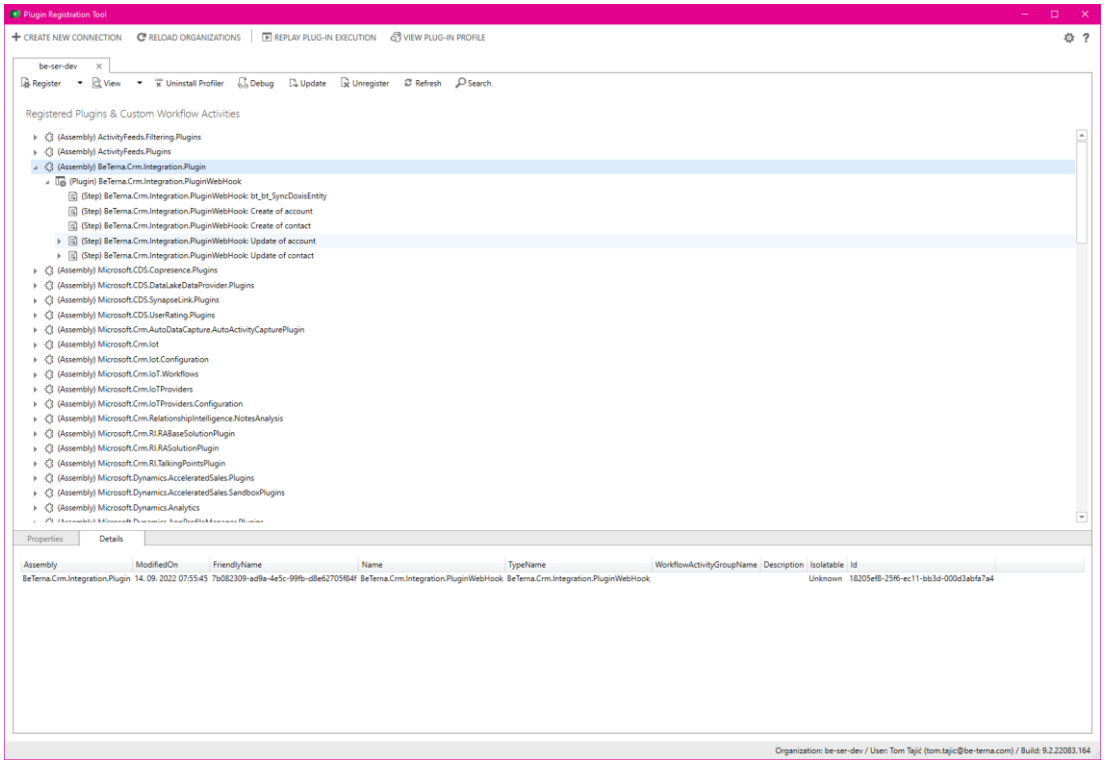
3.5 Setup an entity for automatic sync to Doxis

To enable automatic create / update Doxis synchronization, you have to register your entity for the synchronization.

3.5.1 Step1: Open "Plugin Registration Tool"

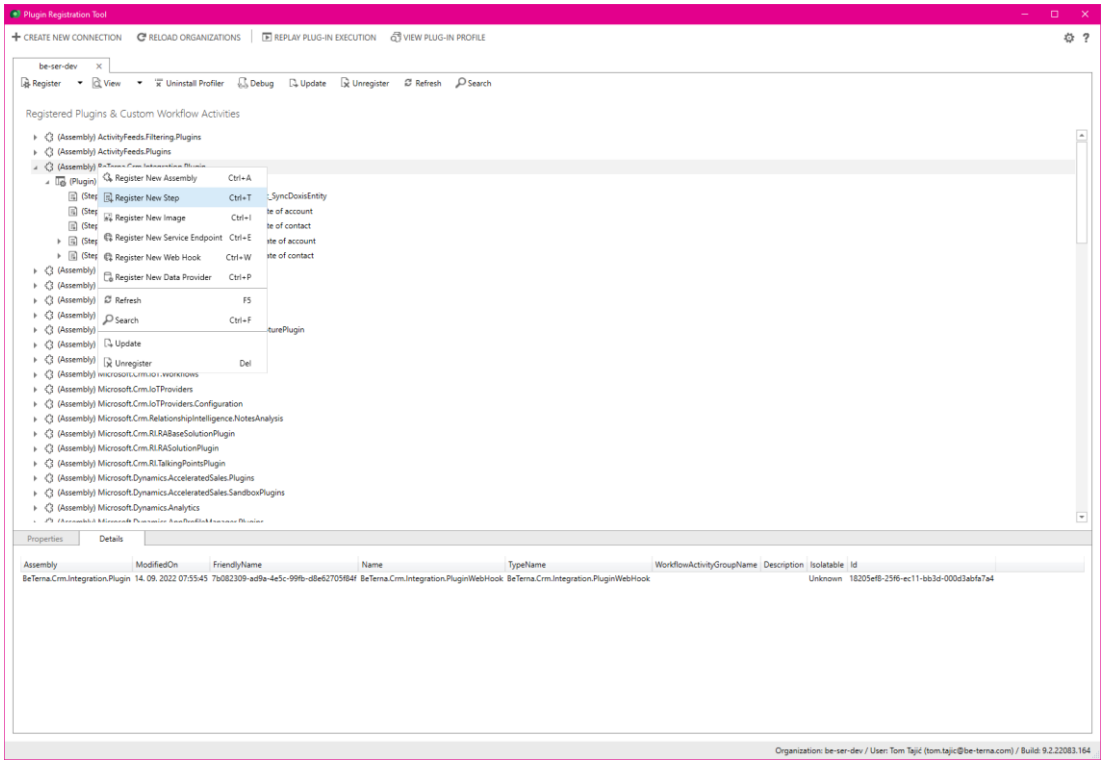
To do this you use the "Plugin Registration Tool". (There is a XrmToolbox plugin or the one that comes with Dynamics SDK).

Look for BeTerna.Crm.Integration.Plugin



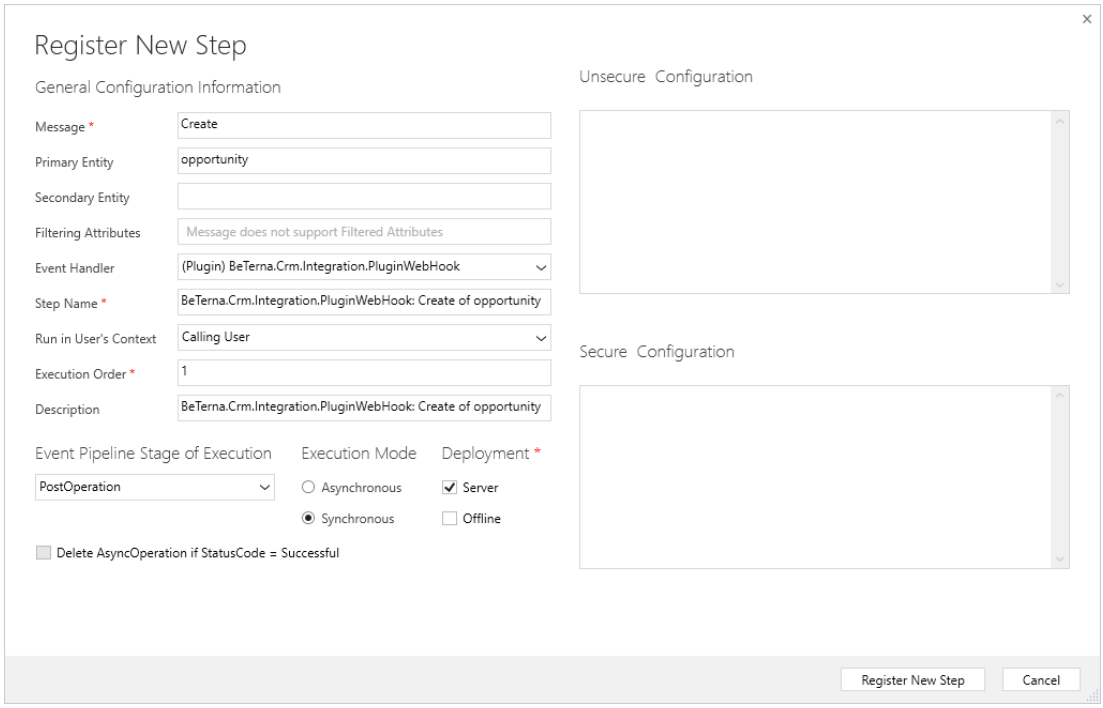
3.5.2 Step2: Register New Step (Create)

Right click on Plugin and select “Register New Step”



Picture 24 Register New Step

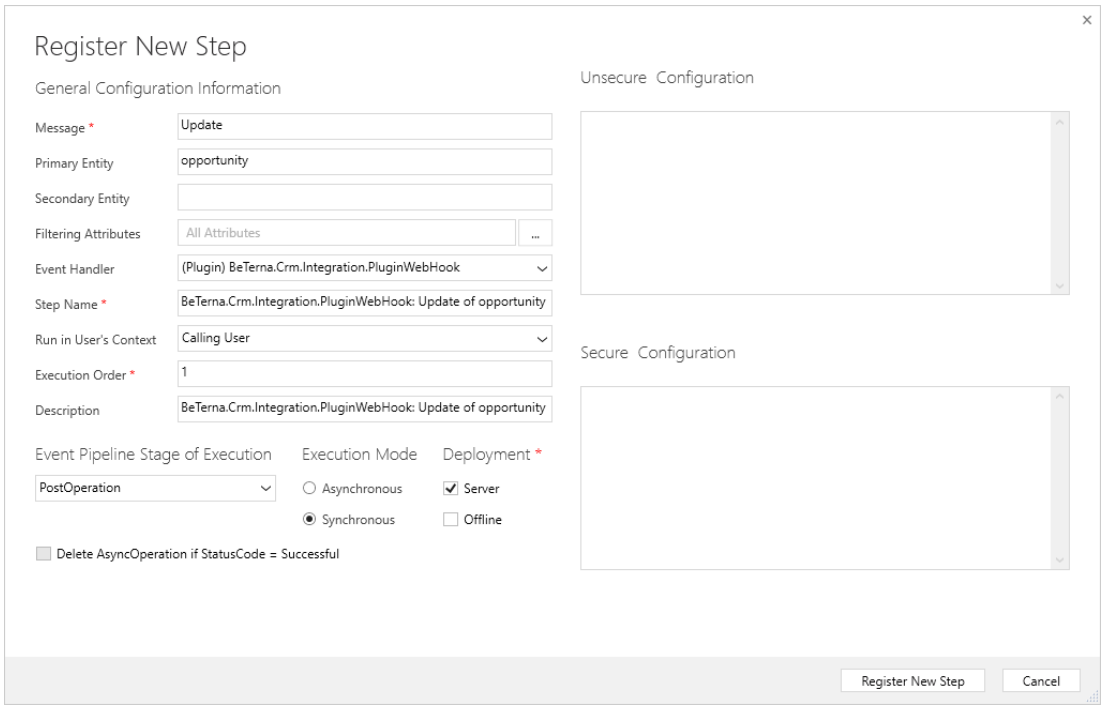
Enter the information below for your entity – replace Primary entity (example is showing "opportunity")



Picture 25 Register your entity ("opportunity" in the example) for Create message with exact same settings

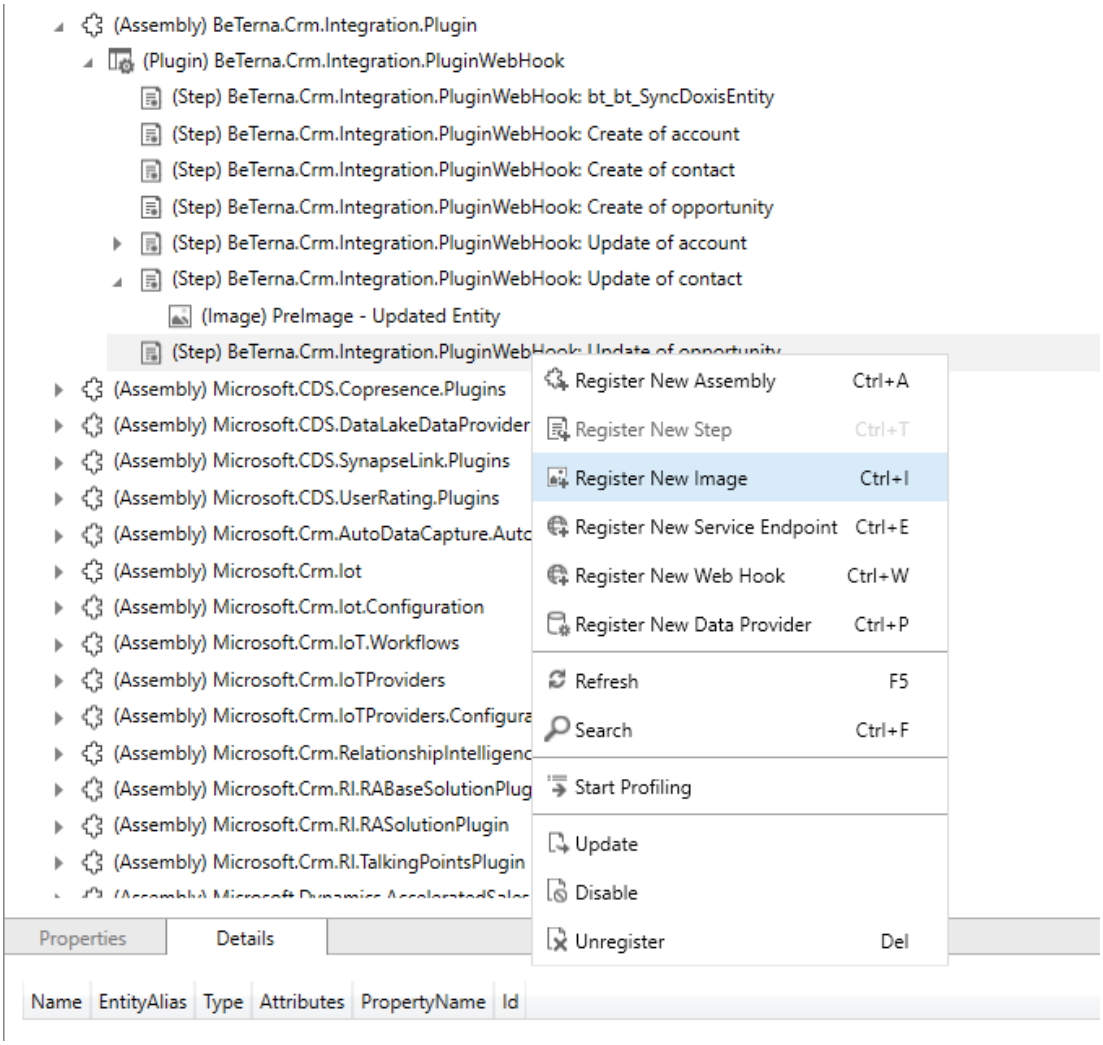
3.5.3 Step3: Register New Step (Update)

Right click on plugin and select "Register New Step".
Enter the information below for your entity – replace Primary entity (example is showing "opportunity")

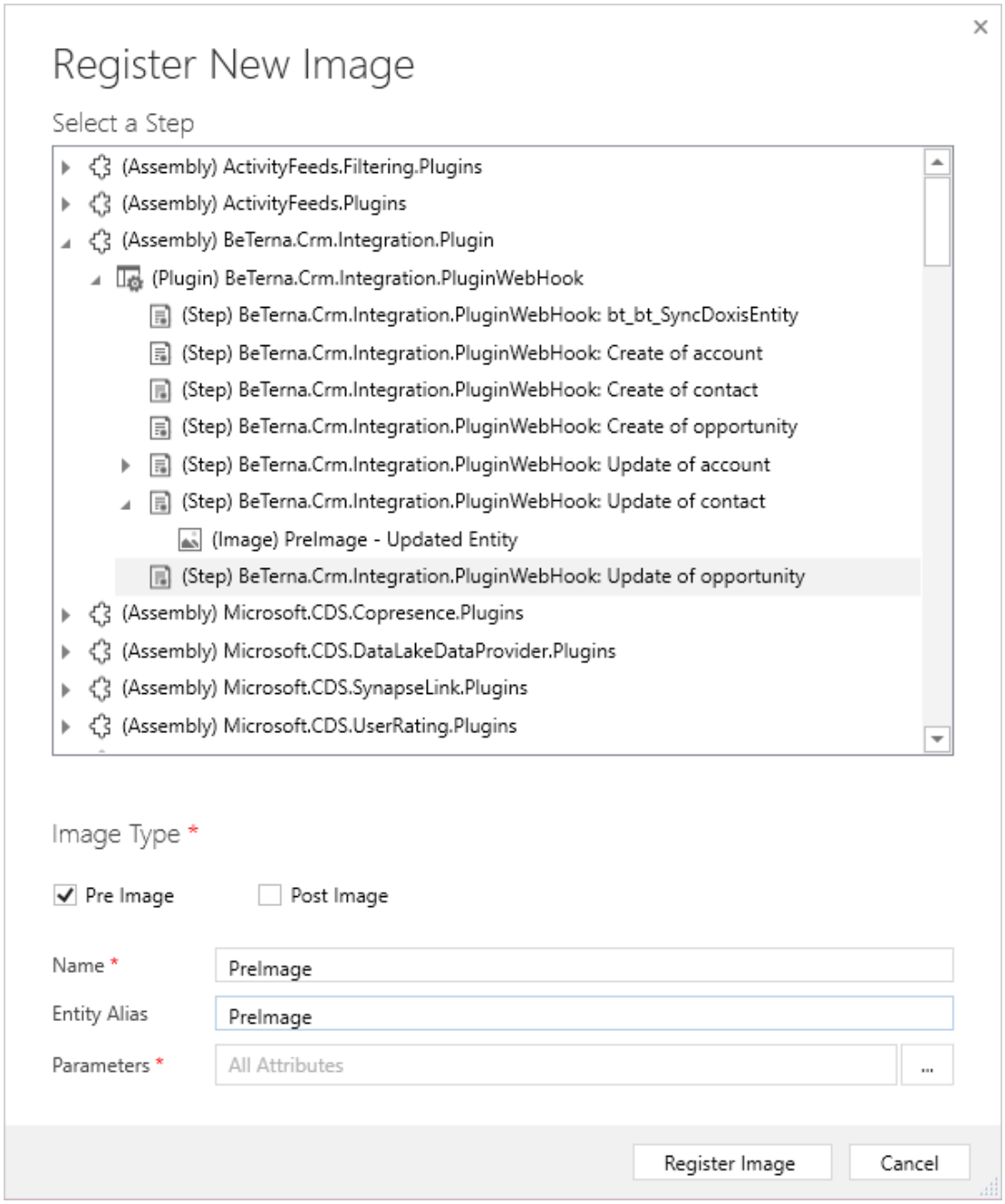


Picture 26 Register your entity ("opportunity" in the example) for Update message with exact same settings

Then you need to register an image for the update step like shown on the picture below. Use the exact same "PreImage" name with the correct letter case.



Picture 27 Register "PreImage" for the Update message



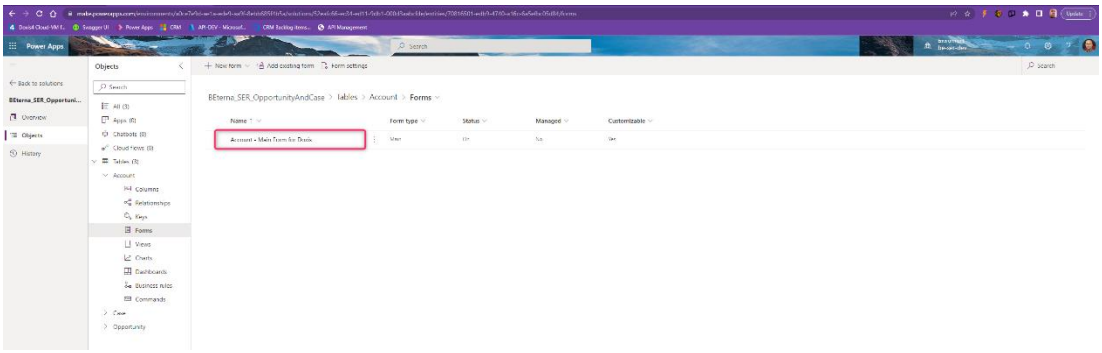
Picture 28 PreImage settings - use exact same settings

3.6 Embed Doxis on entity form

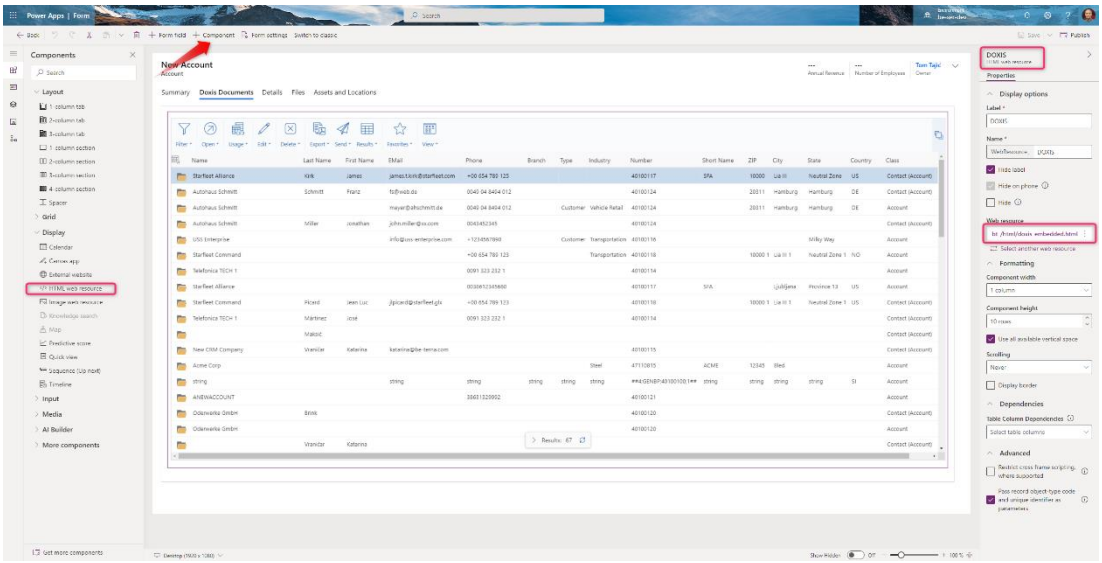
Steps:

1. Go to make.powerapps.com and create a new solution

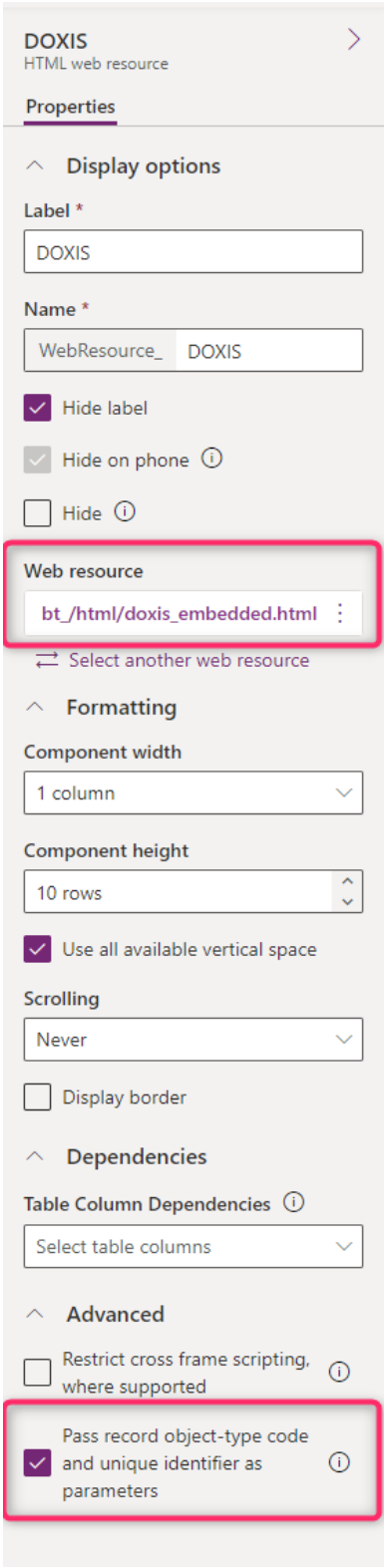
2. Add an entity of your choice and select the form you want to customize / add embedded Doxis frame.
3. Open form designer
4. Add a HTML web resource to the preferred location (+ Component)
5. Configure the HTML web resource as show on the picture below
    - Pick webresource named bt_/html/doxis_embedded.html
    - Make sure the check box "Pass record object-type code and unique identifier as parameters" is selected.
    - This webresource reads the configuration for the current entity and uses the link specified in the "Open DOXIS Url" field of the entity configuration. *If you are customizing a form of an entity for which you have not yet configured the "Open DOXIS Url" field of the entity configuration, then this will not work.*



Add HTML web resource component.



Picture 29 HTML web resource configuration



Picture 30 Mandatory settings of embedded Doxis web resource

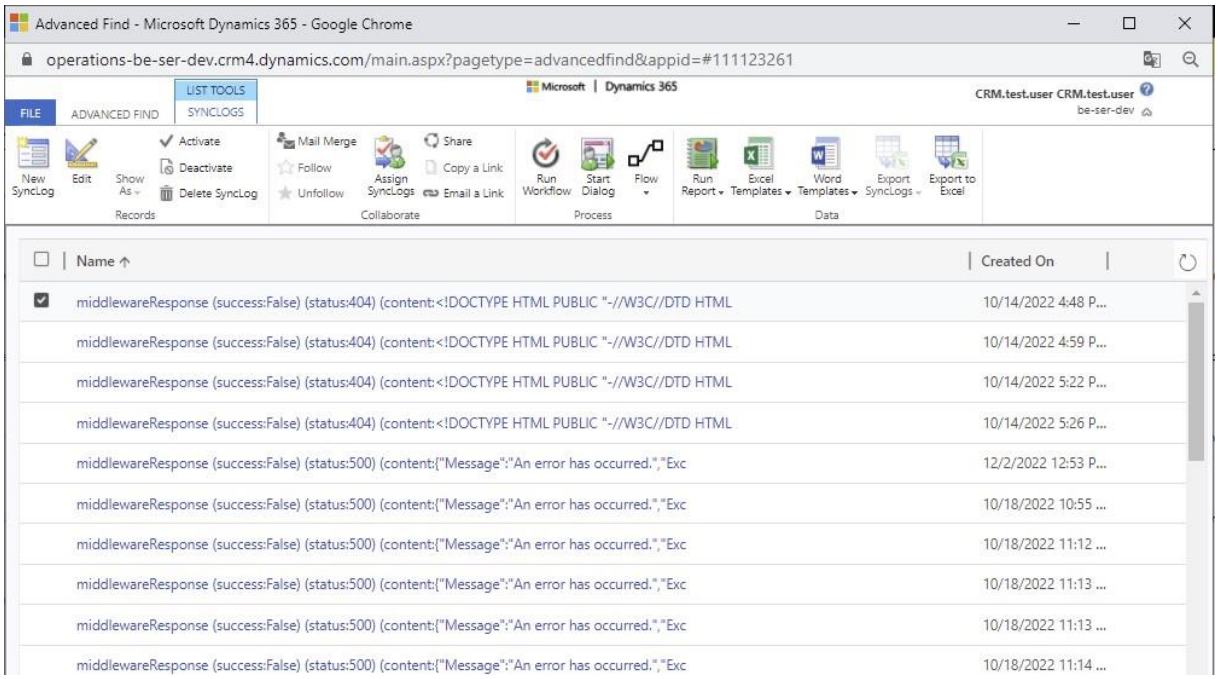
3.7 Inspect logging messages from SmartBridge CRM

When something is not working as expected, you can inspect the messages returned from the middleware SmartBridge CRM in the "SyncLog" (bt_synclog) entity.

To view these messages you have 2 options.

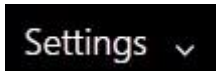
3.7.1 Use advanced find (old CRM implementation)

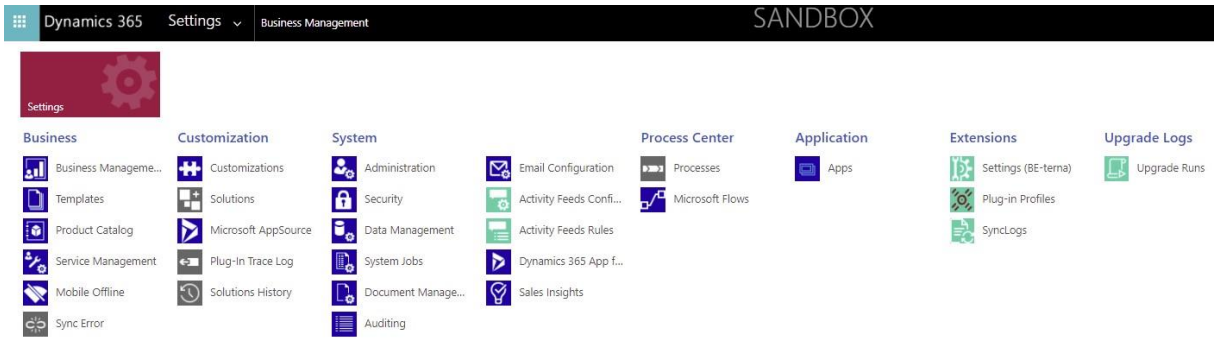
1. Click Settings 
2. Select Advanced Settings
3. Advanced Find 
4. Look for: SyncLogs
5. Results 



Picture 31 SyncLogs - advanced find

3.7.2 Open SyncLogs from old site navigation

1. Click Settings 
2. Select Advanced Settings
3. Settings 
4. Under Extensions click on "SyncLogs"



Picture 32 Sync Logs - site navigation

3.7.3 SyncLog form

SyncLog record contains:

- Source; For now this is always Doxis Sync
- Status Reason; Information, Warning or Error
- Message; depending of where the error occurs the message is structured as

1. Example 1 (error occurred on middleware logic – most probably during the transformation of the message to Doxis json format – indicated you should check your liquid template syntax)

middlewareResponse

(success:False)

(status:500)

(content:{"Message":"An error has occurred.","ExceptionMessage":"Unexpected character encountered while parsing value: C. Path 'ObjectType', line 4, position 16.","ExceptionType":"Newtonsoft.Json.JsonReaderException","StackTrace":" at Newtonsoft.Json.JsonTextReader.ParseValue()\r\n at

Newtonsoft.Json.JsonTextReader.Read()\r\n at Newtonsoft.Json.Linq.JContainer.ReadContentFrom(JsonReader r, JsonLoadSettings settings)\r\n at Newtonsoft.Json.Linq.JContainer.ReadTokenFrom(JsonReader reader, JsonLoadSettings options)\r\n at Newtonsoft.Json.Linq.JObject.Load(JsonReader reader, JsonLoadSettings settings)\r\n at Newtonsoft.Json.Linq.JObject.Parse(String json, JsonLoadSettings settings)\r\n at BeTerna.SER.CRM.WebHooks.Functions.PluginWebHook.Run(HttpRequest req, ILogger log) in C:_CODE_PROJECTS\\SER\\Solution\\BeTerna.SER.CRM\\BeTerna.SER.CRM.WebHooks\\Functions\\PluginWebHook.cs:line 96"})

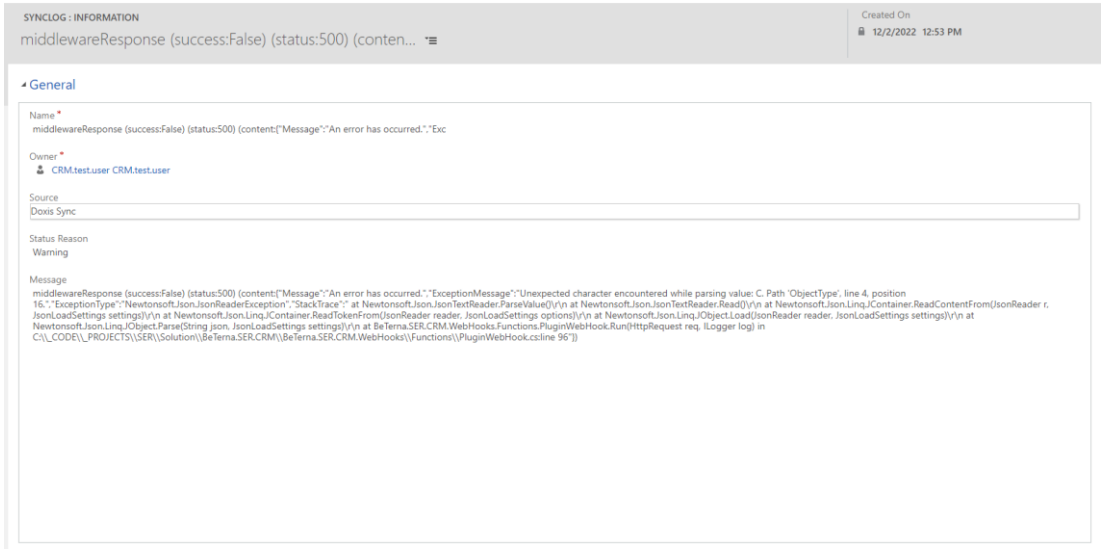
2. Example 2 (error occurred on Doxis API – in this example value sent to Doxis was not in an expected range):

middlewareResponse

(success:True)

(status:200)

(content:{"StatusCode":400,"Content":"{\"status\":400,\"message\":\"Values other than enumerated: ObjectState\"}","Message":""})



Picture 33 SyncLog - form