**Detailed Design Document:**

**SPA Web Service interface**

|  |  |  |  |
| --- | --- | --- | --- |
| **Main Changes** | **Date** | **Editor** | **Version** |
| First Draft | 27/11/2019 | Yves TAYAR | 1.0 |

Contents

[1. General 2](#_Toc25752031)

# General

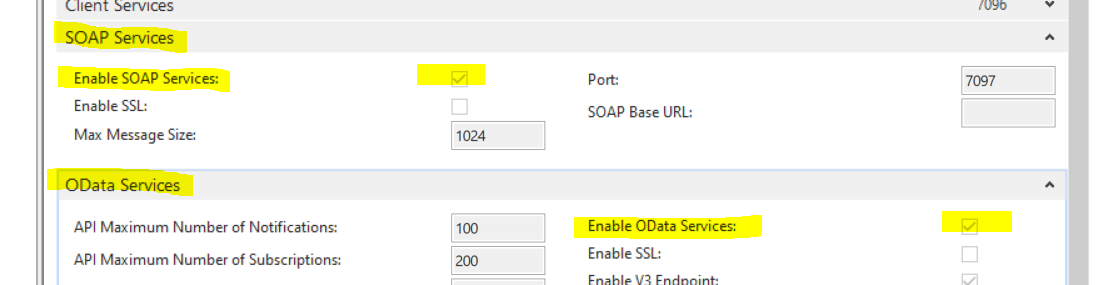
This guide will describe how to implement web service interface:

* From Nav create Code
* To Nav : Ho to access to web service

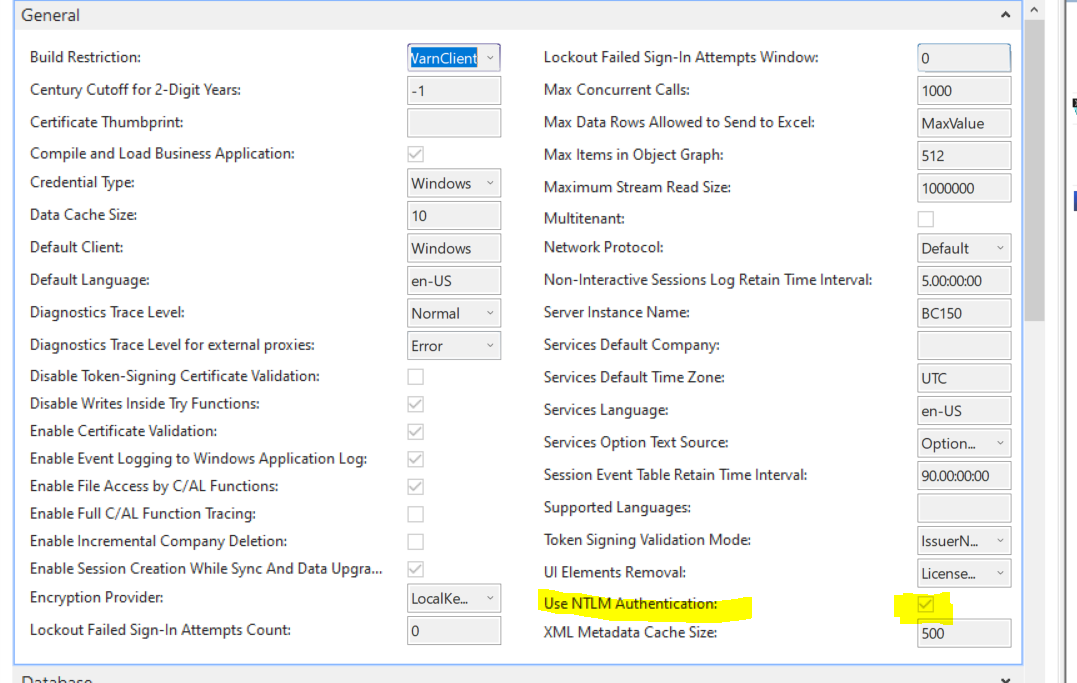
# Setup:

On NAV Business Central Administration :

Activate Odata + SOAP web Service



On General Tab Activate option : "use NTLM Authentication"



Then restart the service

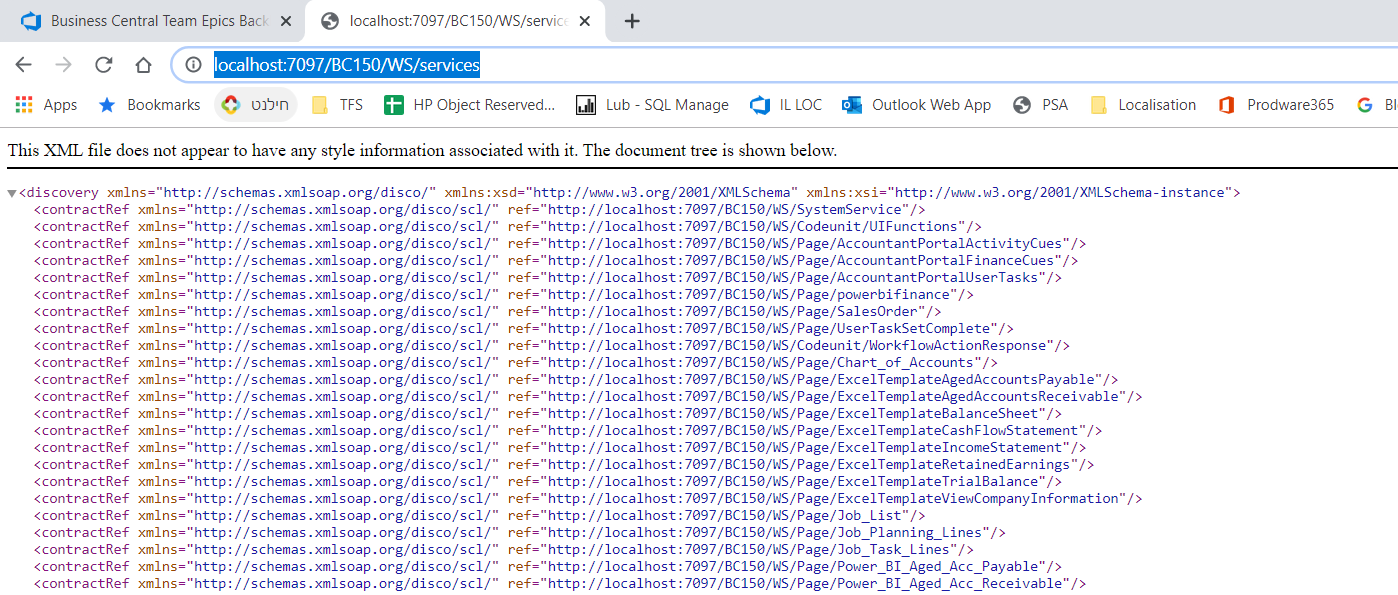
# Check Web Service access:

To test the web service access Open Browser and Fill address:

<http://server:SoapPort/Instancename/WS>/services

on our Exemple : <http://localhost:7097/BC150/WS/services>

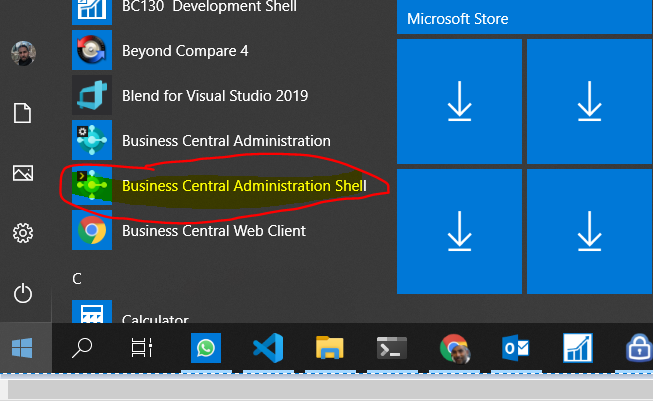
The result should be the list of all available functions :



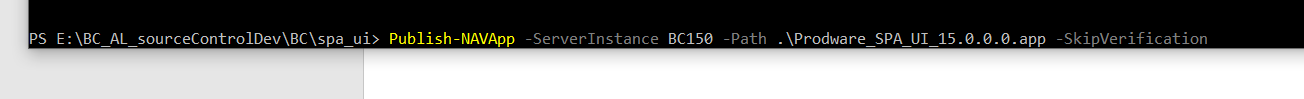
# Publish and Install App :

Copy the extension to a local folder

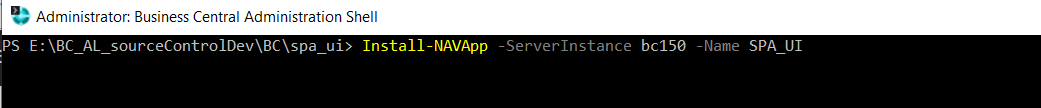
Open Business Centrall Administration Shell (**Run As administrator**)

Open 

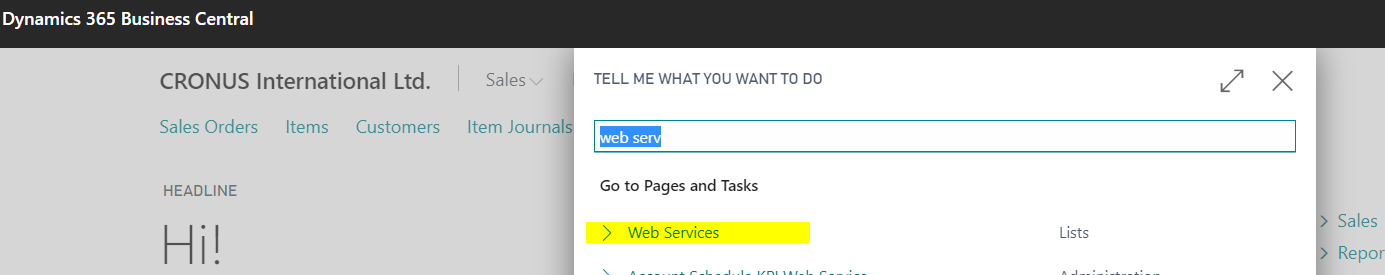
Go to the folder that exist the extension and publish



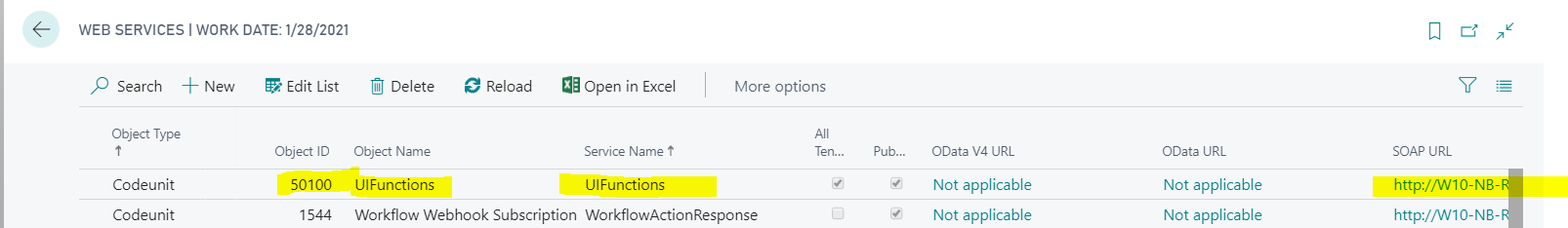
And then Install the App



Open Business central and search "Web service" Page



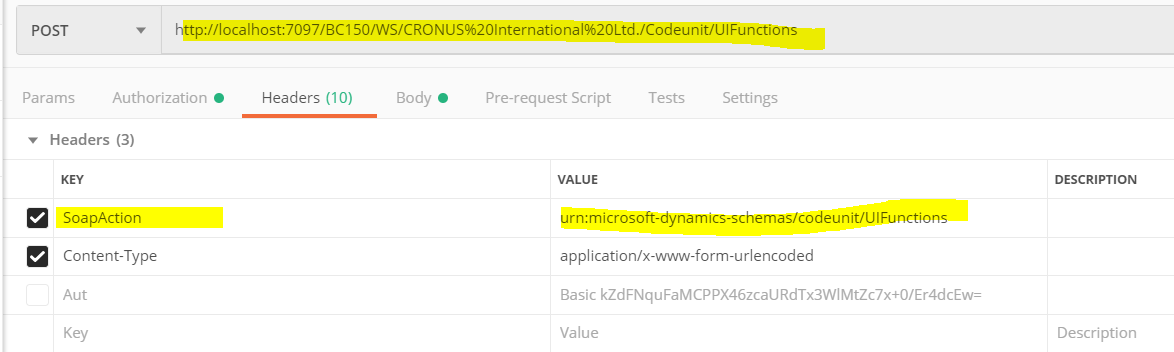
Create a new line exactly as is:



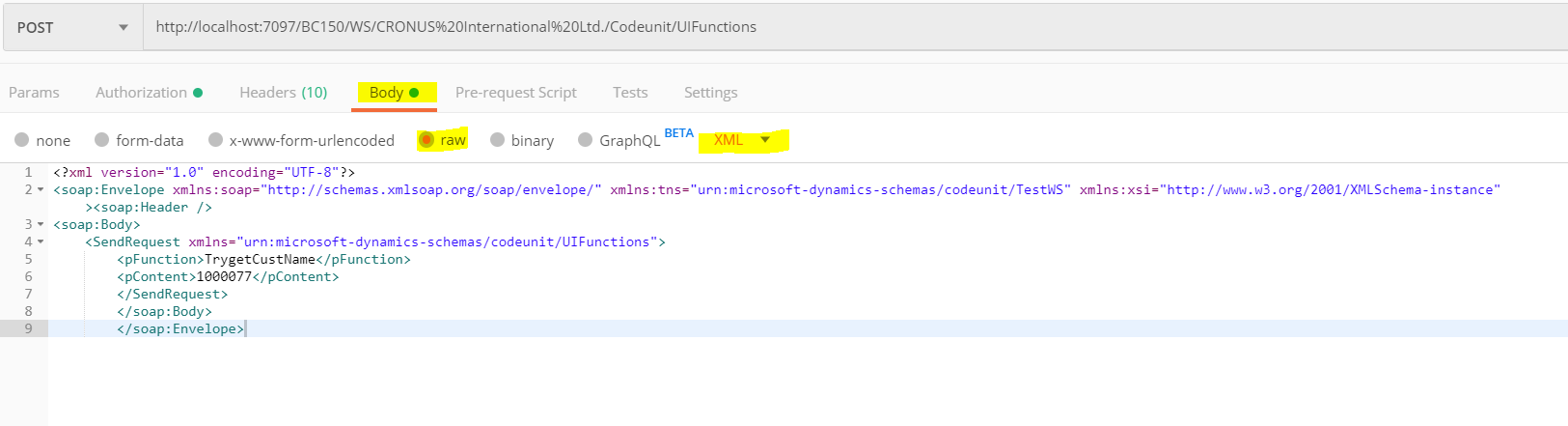
# Run on Postman the test function

Create a new definition

Important Fill The SoapAction Header



On the body :



<?xml version="1.0" encoding="UTF-8"?>

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:tns="urn:microsoft-dynamics-schemas/codeunit/TestWS" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<soap:Header />

<soap:Body>

<SendRequest xmlns="urn:microsoft-dynamics-schemas/codeunit/UIFunctions">

<pFunction>TrygetCustName</pFunction>

<pContent><![CDATA[1000077]]></pContent>

</SendRequest>

</soap:Body>

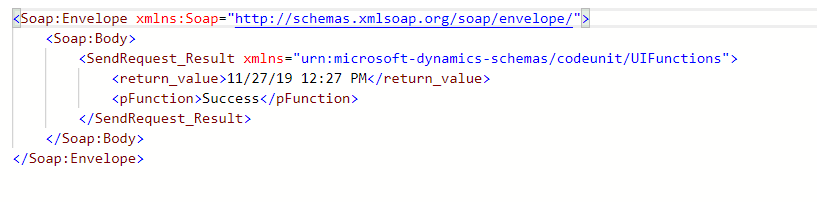
</soap:Envelope>

The Body Should send 2 Parameter :

pFunction wich is the function to run

pContent wich is the content, it can be an XML / JSON / or simply Text (in order to preserve the format the message should be with the <![CDATA[ prefixe when content is xml or json)

the result should be :



If success the value of parameter pFunction will be "Success"

If not the value of parameter pFunction will be "error" and the return\_value should be the error description

# In Business Central AL Code create Function

In Visual Studio Create a Extension an add the API\_UI extension as dependancie

The create a Subscriber that will receive 2 Parameters on UIFunctions Codeunit :

The pFunction will then function

    //Suscriber for test Purpose

    [EventSubscriber(ObjectType::Codeunit, Codeunit::UIFunctions, 'WSPublisher', '', true, true)]

    local procedure FirstSubscriber(VAR pFunction: Text[50]; VAR pContent: Text)

    VAR

        Customer: Record "Customer";

    begin

        IF pFunction <> 'TrygetCustName' THEN

            EXIT;

        Customer.GET(pContent);

        Customer.Name := FORMAT(CURRENTDATETIME);

        Customer.MODIFY;

        pContent := Customer.Name;

    end;