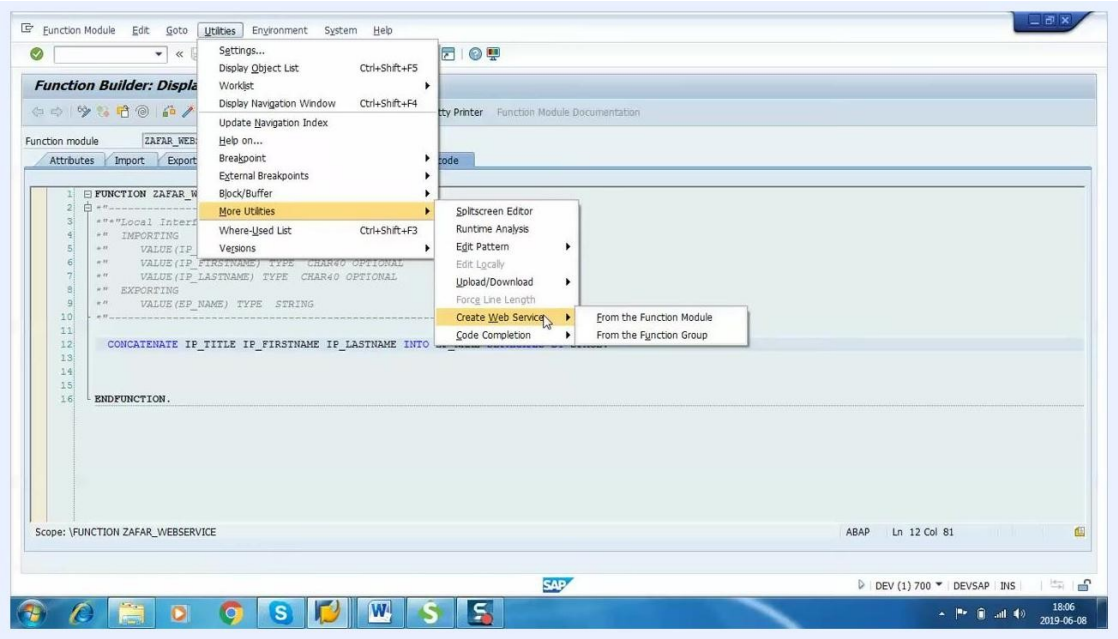
How to Create a Web Service in SAP ABAP

Creating a web service in SAP ABAP lets you expose business logic built in a function module so that external systems can consume it over SOAP. This guide walks through the full process: creating a function group, building a function module, generating a web service definition, configuring the service binding in SOAMANAGER, exporting the WSDL, and testing the finished web service with SoapUI.



The SAP Function Builder screen is open with the 'Utilities' menu expanded. The cursor is hovering over 'More Utilities', revealing a submenu with options including 'Create Web Service' and 'Code Completion'

Prerequisites

Before you begin, make sure you have:

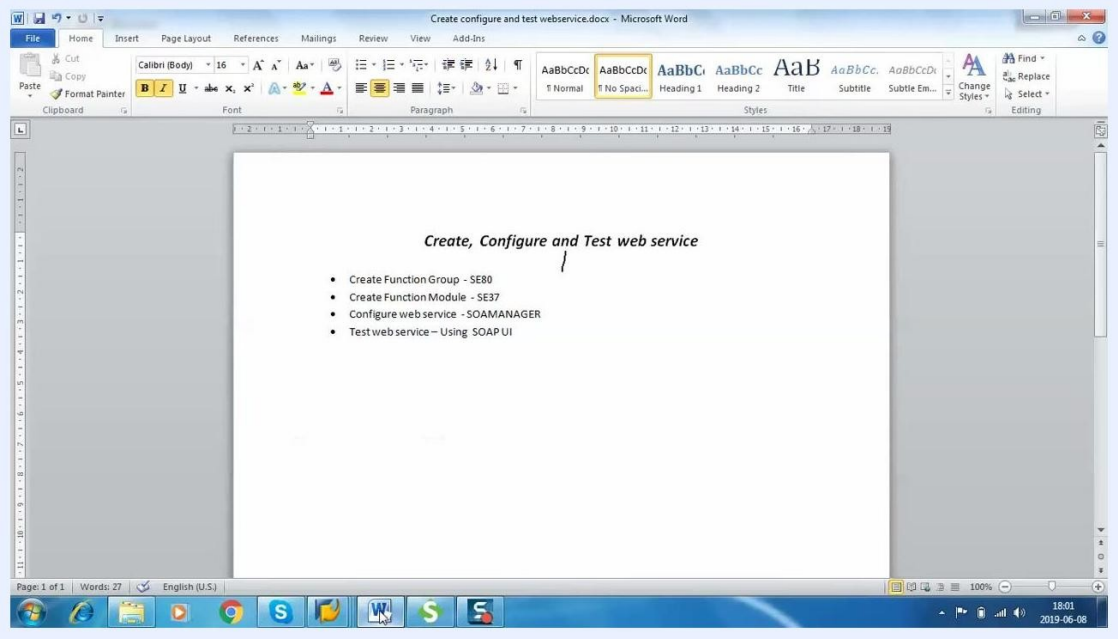
- Access to the SAP GUI with developer authorizations (SE80, SE37, SOAMANAGER)
- A package to assign your objects to (for example, zabap1 or ZABAP)
- Access to the SOA Manager URL in a web browser
- SoapUI installed on your machine for testing the finished web service

The overall process consists of four main stages: creating the function group in **SE80**, creating the function module in **SE37**, configuring the web service in **SOAMANAGER**, and testing the web service using **SoapUI**.

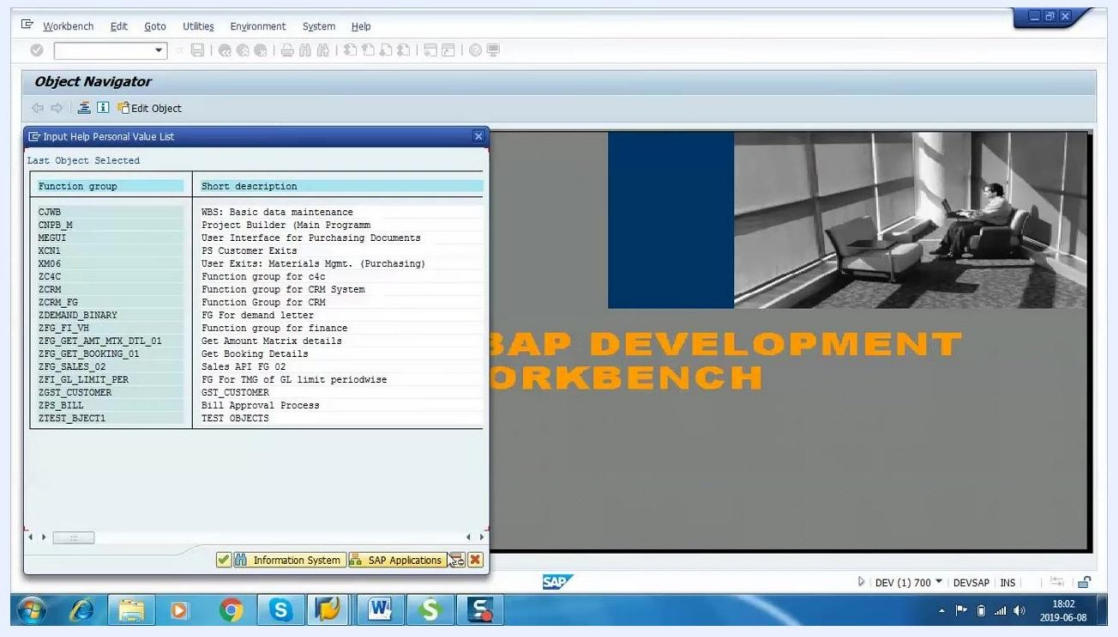
Step 1: Create the function group

1. Open the SAP GUI and navigate to the ABAP Development Workbench (**SE80**).
2. In the left panel, select **Function Group** from the dropdown menu.
3. Enter the name for the new function group. For example, enter ZAFAR_FG.
4. Press **Enter**.

5. If the function group does not exist, a prompt appears: "Function Group ZAFAR_FG does not exist. Create Object?" Click **Yes** to proceed.



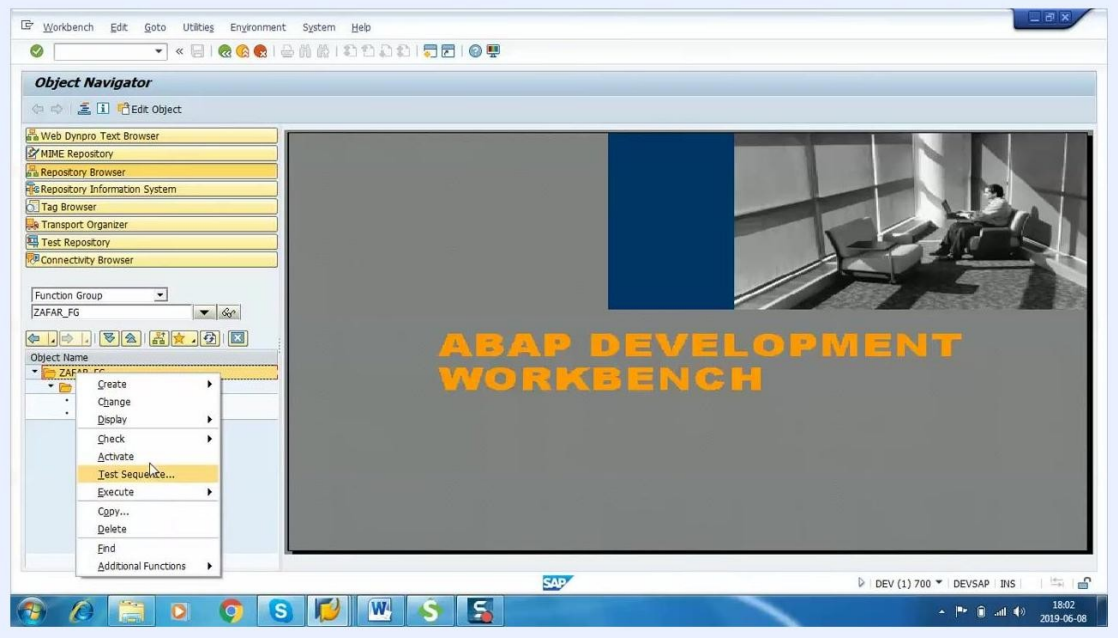
Microsoft Word document listing the four main steps: Create Function Group – SE80, Create Function Module – SE37, Configure web service – SOAMANAGER, Test web service – using SoapUI



SAP ABAP Development Workbench with 'Function Group' selected in the dropdown and the name ZAFAR_FG entered

In the **Create Object Directory Entry** dialog:

- Enter a short description, such as "Text for web service."
- For the package, enter zabap1.
- Set **Person Responsible** to DEVELOPER1.
- Confirm the original system and language as shown.
- Click **Local Object**, or assign the object to a package as needed.

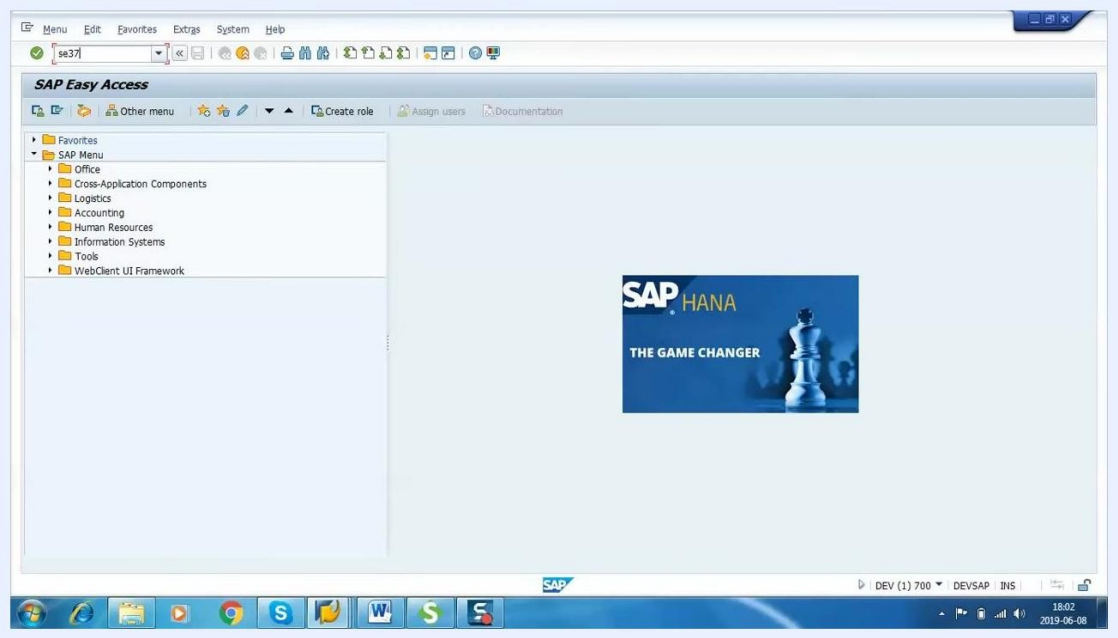


Right-click context menu on the function group ZAFAR_FG, showing options including Activate

Once the function group is created, right-click on the function group name (ZAFAR_FG) in the object list and select **Activate** from the context menu to activate the function group.

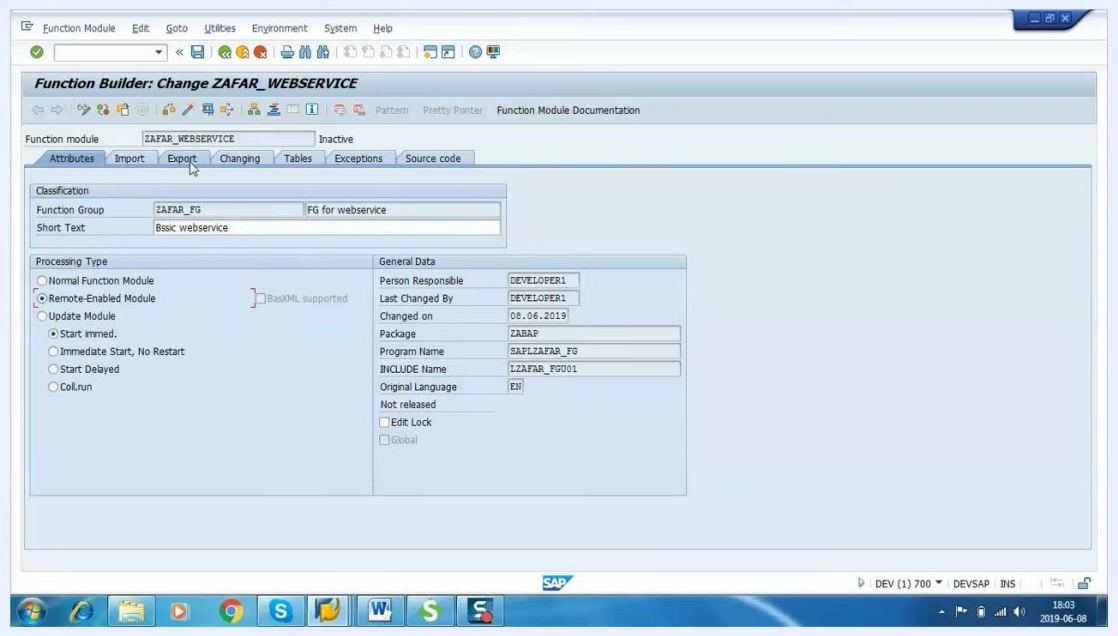
Step 2: Create the function module

1. Open the SAP GUI and enter transaction code **SE37** in the command field, then press **Enter** to navigate to the Function Builder.



SAP Easy Access screen with 'se37' entered in the command field, ready to access the Function Builder

2. In the **Function Builder: Initial Screen**, enter the function module name ZAFAR_WEBSERVICE, then click **Create**.

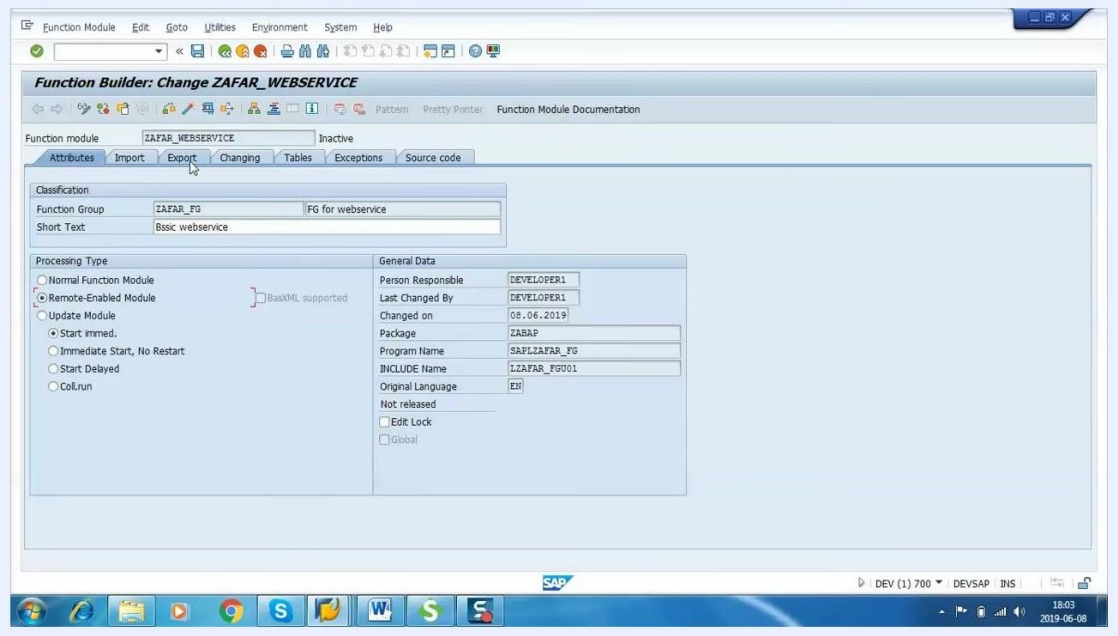


Function Builder: Initial Screen with ZAFAR_WEBSERVICE entered as the function module name and the Create button visible

3. In the **Create Function Module** dialog, enter the following details:

   - **Function Module:** ZAFAR_WEBSERVICE
   - **Function Group:** ZAFAR_FG

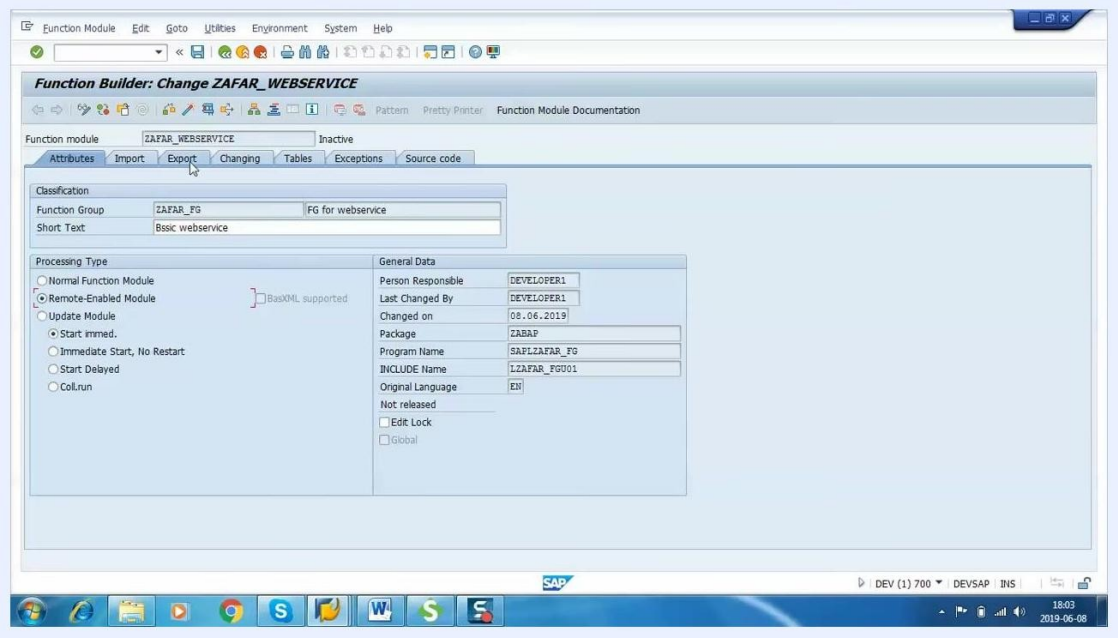
- **Short Text:** "Basic webservice"



Create Function Module dialog with ZAFAR_WEBSERVICE and ZAFAR_FG filled in, and a warning about reserved names

If a warning appears stating that the function module name is reserved for SAP, click **New entry** to proceed.

4. In the **Function Builder: Change** screen, open the **Attributes** tab and set **Processing Type** to **Remote-Enabled Module** by selecting the corresponding radio button. Confirm the function group, short text, and other general data as shown.

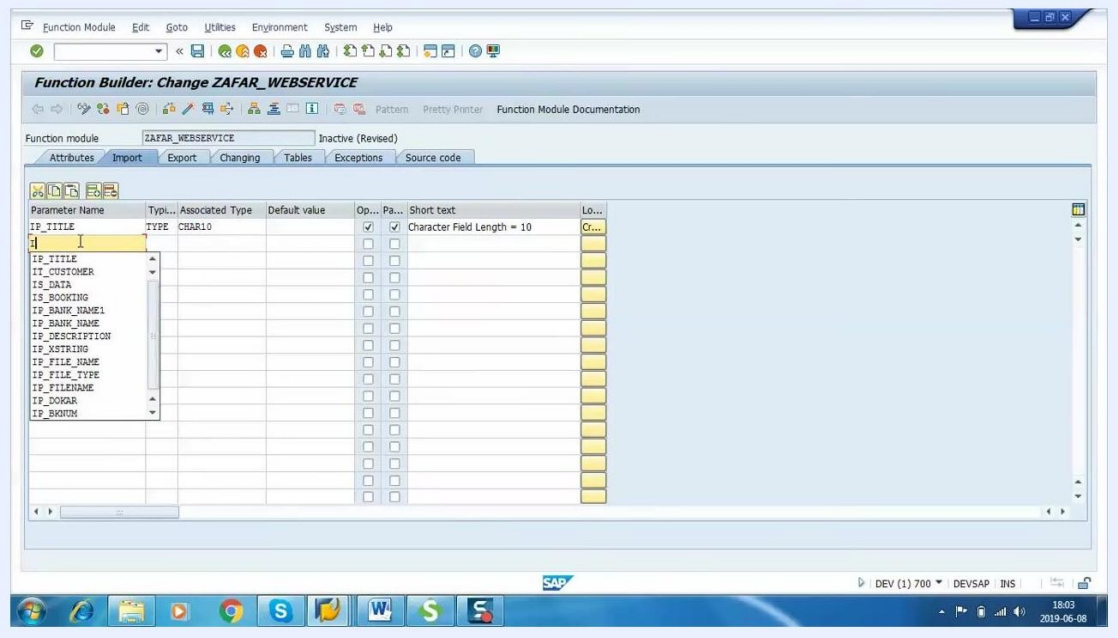


Function Builder: Change screen with Attributes tab open, Remote-Enabled Module selected, and details filled in

5. Go to the **Import** tab and add the following parameters:

   - IP_TITLE of type CHAR10 (Character Field Length = 10)
   - IP_FIRSTNAME of type CHAR40 (Character Field Length = 40)

- IP_LASTNAME of type CHAR40 (Character Field Length = 40)



Import tab with IP_TITLE, IP_FIRSTNAME, and IP_LASTNAME parameters defined, and an information dialog about RFC parameters

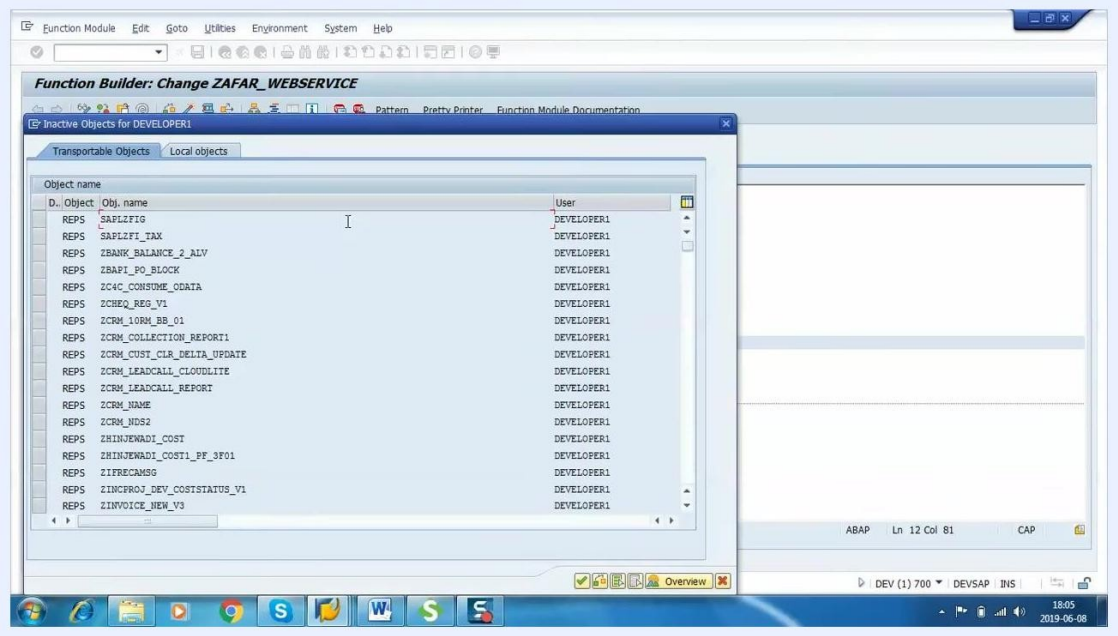
If an information dialog appears stating that reference parameters are not allowed with RFC, acknowledge it and ensure all parameters use simple data types.

6. Go to the **Export** tab and add the following parameter:

    - EP_NAME of type STRING

Step 3: Implement and test the function module logic

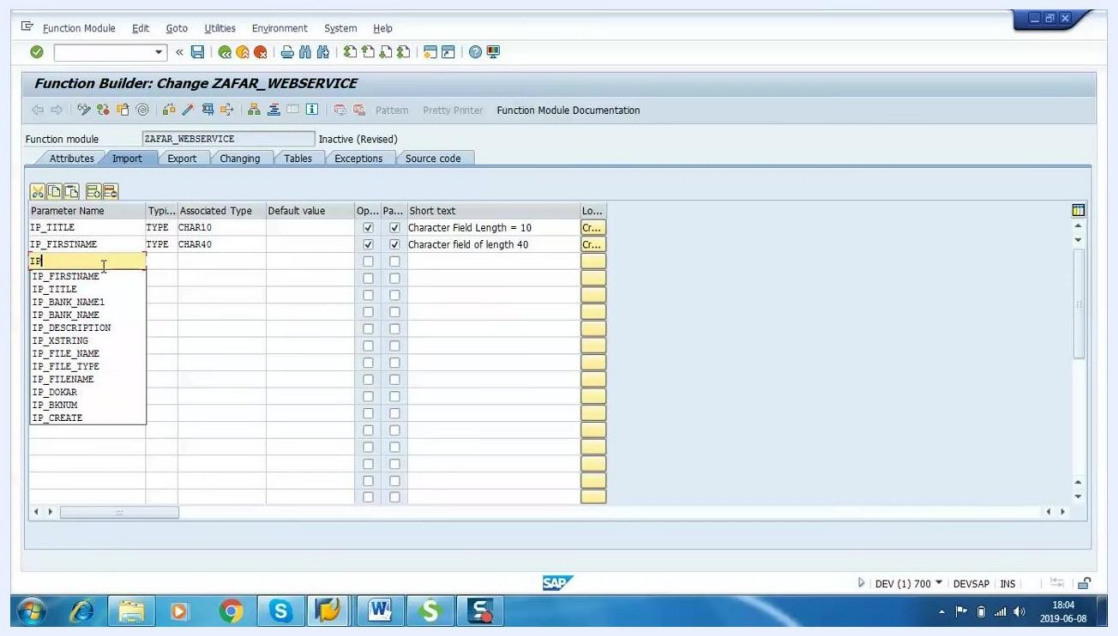
1. In the **Source code** tab of the Function Builder for ZAFAR_WEBSERVICE, enter the ABAP code to concatenate the input parameters:



Source code tab in Function Builder with CONCATENATE statement being typed for IP_TITLE, IP_FIRSTNAME, and IP_LASTNAME

```
  CONCATENATE IP_TITLE IP_FIRSTNAME IP_LASTNAME INTO EP_NAME
SEPARATED BY SPACE.
```

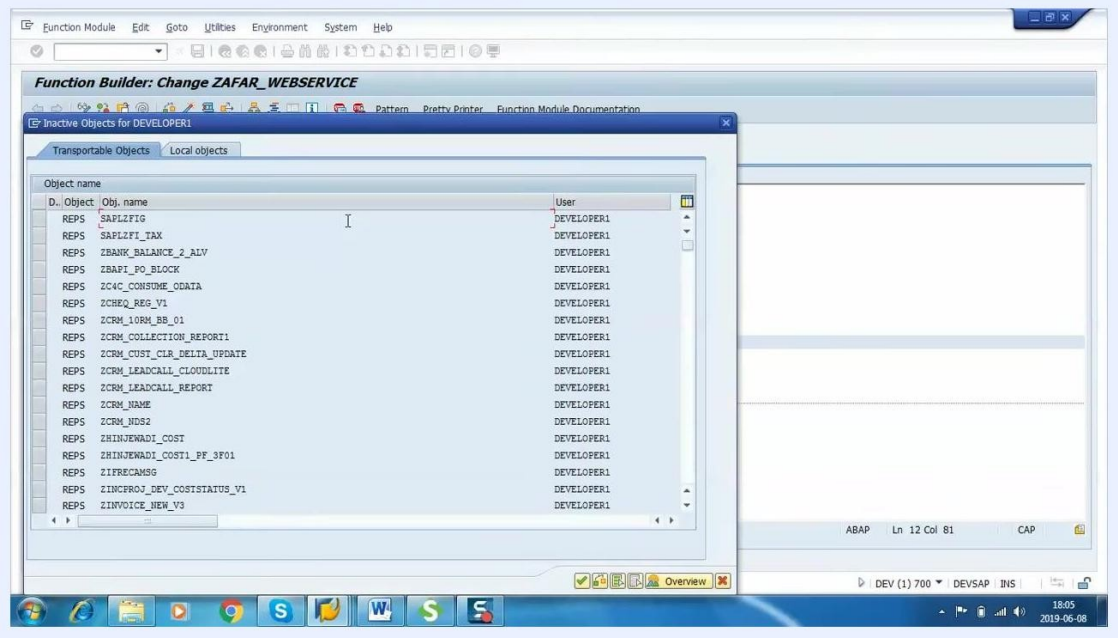
This code combines the values of IP_TITLE, IP_FIRSTNAME, and IP_LASTNAME into the export parameter EP_NAME, separated by spaces.



Import tab with all three parameters (IP_TITLE, IP_FIRSTNAME, IP_LASTNAME) defined and ready to add export parameter

2. Save and activate the function module. Confirm the status at the top changes to "Active".
3. Click the **Test/Execute** (F8) button in the Function Builder toolbar. The **Test Function Module: Initial Screen** appears.
4. Enter test values for the import parameters:
    - **IP_TITLE:** Mr.
    - **IP_FIRSTNAME:** Zafar

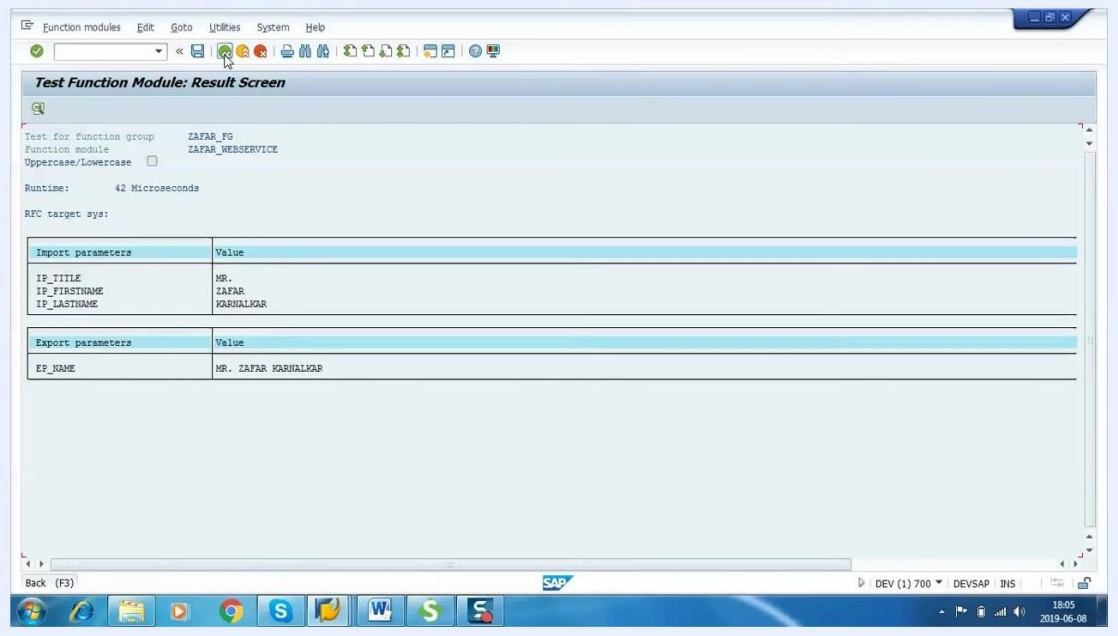
- **IP_LASTNAME:** Karnalkar



Test Function Module: Initial Screen with IP_TITLE=Mr., IP_FIRSTNAME=Zafar, IP_LASTNAME=Karnalkar

5. Execute the test. The result screen displays the concatenated output in the export parameter:

- **EP_NAME:** MR. ZAFAR KARNALKAR

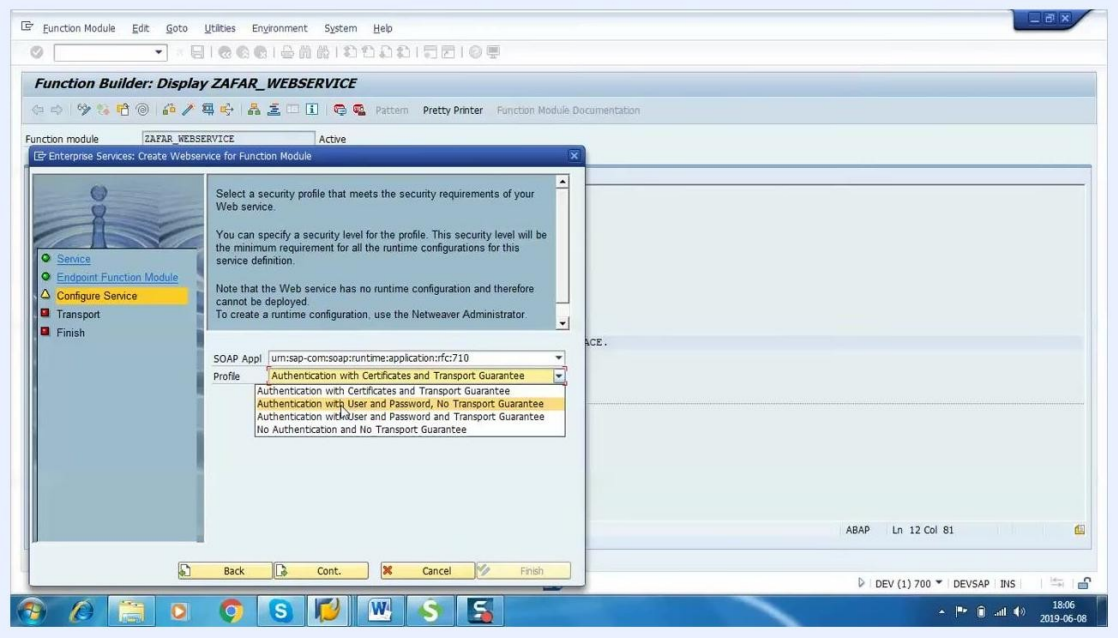


Test Function Module: Result Screen showing input parameters and export parameter EP_NAME = MR. ZAFAR KARNALKAR

Step 4: Create a web service from the function module

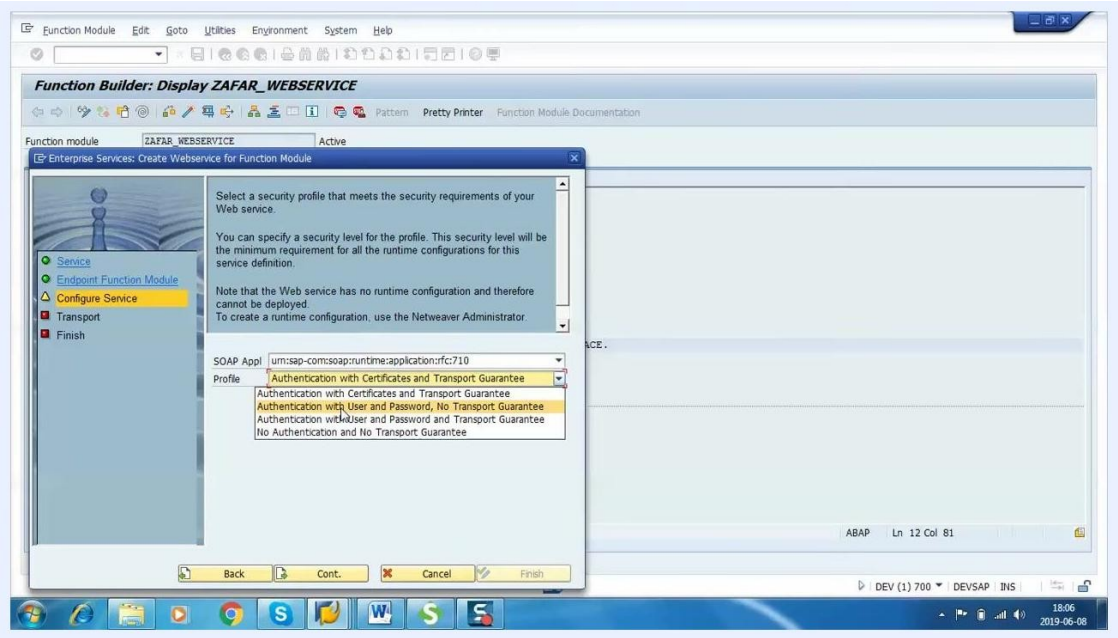
1. Make sure you are in the Function Builder screen with your target function module (ZAFAR_WEBSERVICE) displayed.
2. In the menu bar, click **Utilities**, then hover over **More Utilities** to expand additional options.
3. In the expanded menu, hover over **Create Web Service** and choose **From the Function Module** from the next submenu.

4. In the **Enterprise Services: Create Webservice for Function Module** dialog, enter a name for your web service in the **Service Definition** field (for example, ZWS_NAME). Optionally, provide a description for the web service, then click **Cont.** (Continue) to proceed.



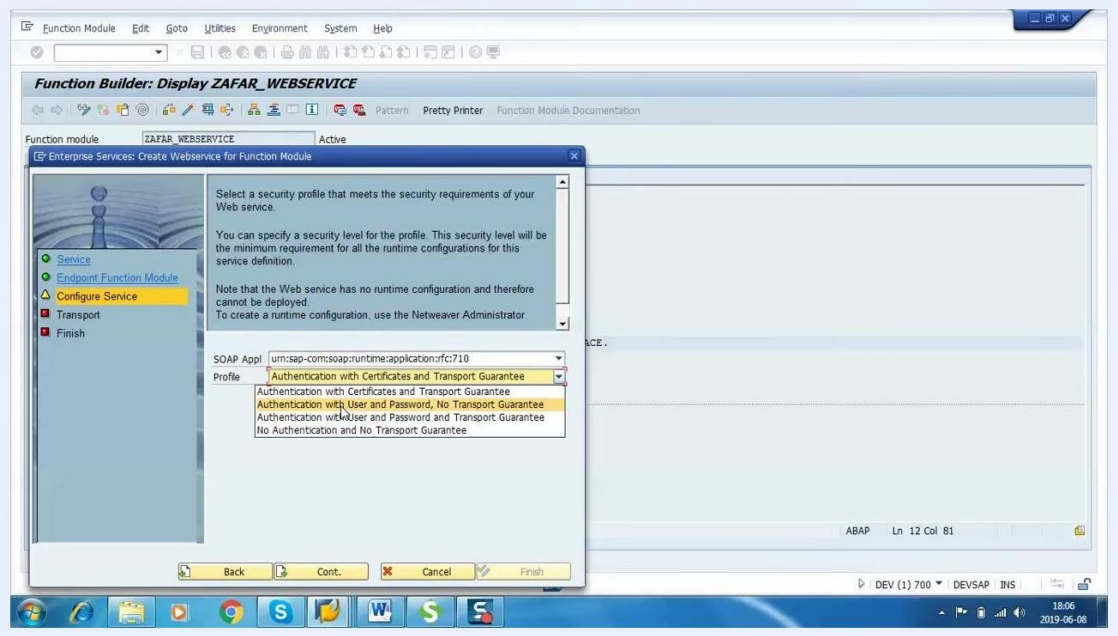
Dialog for creating a web service is open. The user is entering 'ZWS_' as the service definition, with suggestions like 'ZWS_EMAIL_RAW_DATA' visible. The description field is empty

5. In the **Configure Service** step, select the appropriate SOAP Application URI (for example, urn:sap-com:soap:runtime:application:rfc:710). In the **Profile** dropdown, choose the desired security profile, such as **Authentication with Certificates and Transport Guarantee**.



Dialog for configuring the web service is open. The SOAP Application URI is set, and the profile dropdown is expanded, showing options like 'Authentication with Certificates and Transport Guarantee', 'Authentication with User and Password, No Transport Guarantee', etc.

6. Continue through the wizard until you reach the **Finish** step. Review the summary, which confirms the web service (ZWS_NAME) will be created, then click **Complete** to finalize the creation.

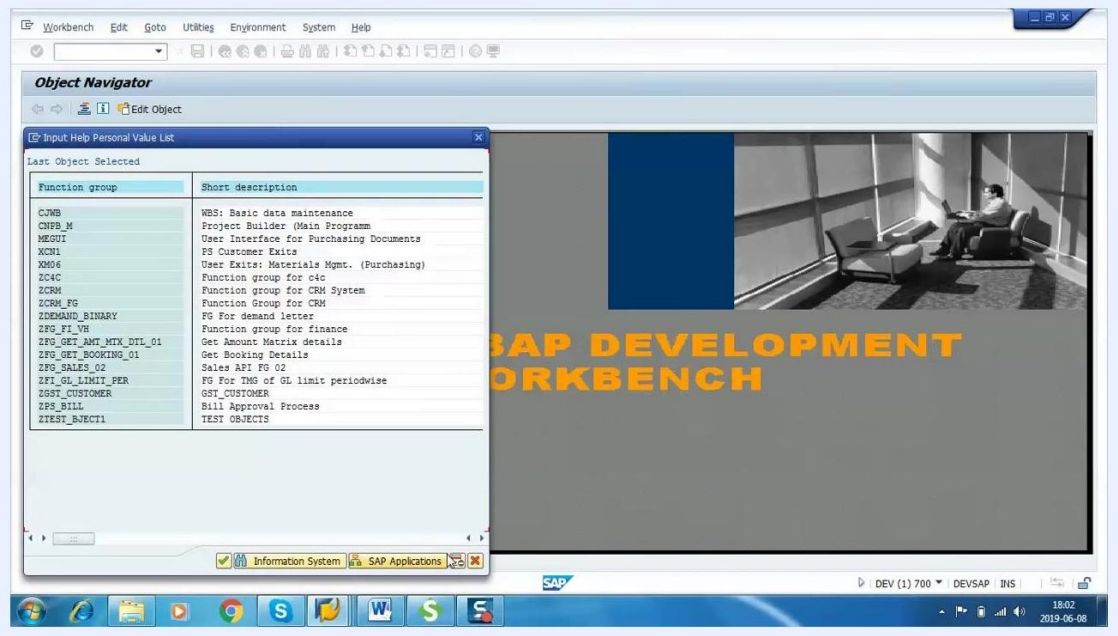


The final step of the wizard is shown, with instructions to click 'Finish' to create the web service ZWS_NAME. The summary notes that runtime release must be created in SOAMANAGER

7. The system displays the **Change Service Definition** screen for your new web service. Review the details:

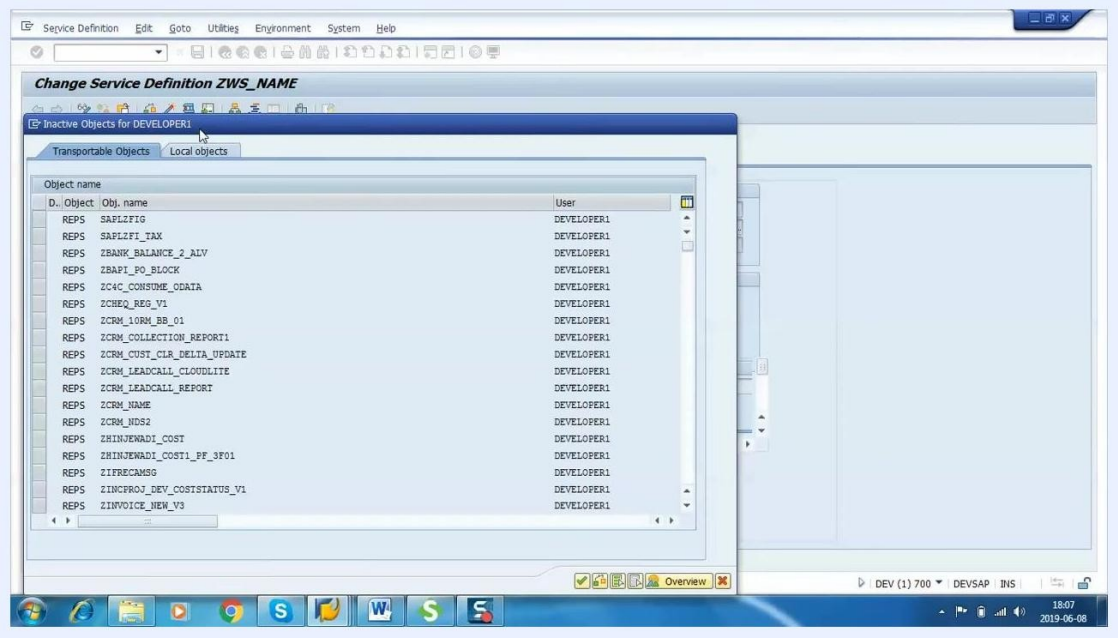
   - **Technical Name:** ZWS_NAME
   - **Porttype Name:** ZWS_NAME
   - **Short Description:** "Basic web service for test"
   - **SOAP Namespace:** urn:sap-com:document:sap:rfc:functions
   - **Endpoint Type:** Function Module
   - **Referenced Object:** ZAFAR_WEBSERVICE
   - **Package:** ZABAP

- **Created/Changed by:** DEVELOPER1



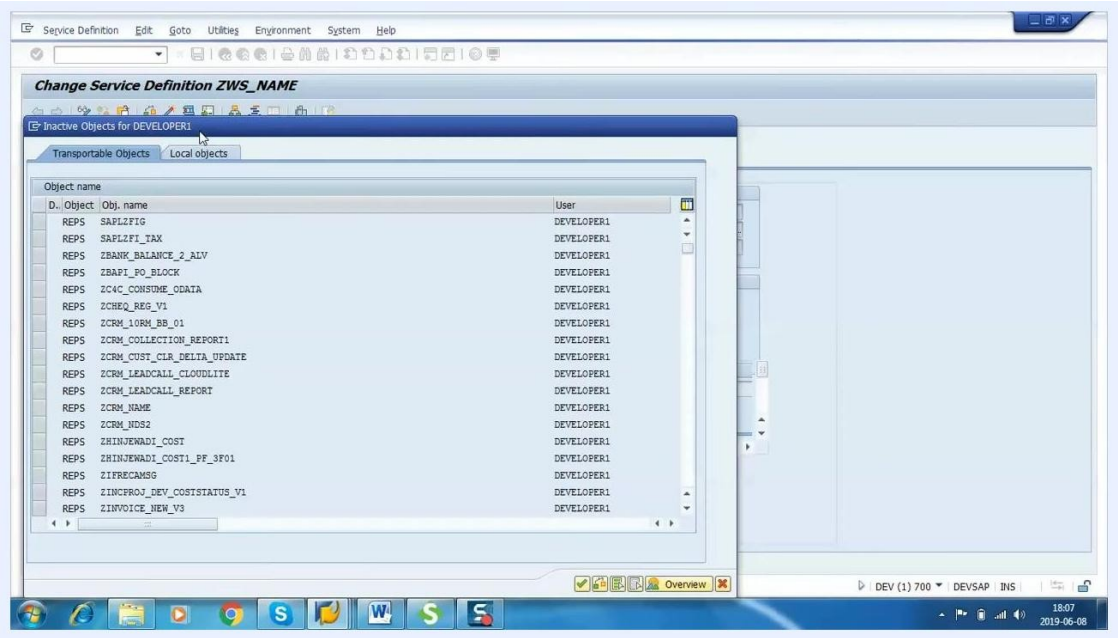
Dialog for creating a function group, with package set to zabap1 and person responsible set to DEVELOPER1

8. If prompted, select the appropriate transport request for your changes. The **Inactive Objects for DEVELOPER1** dialog may appear, listing objects and users. Choose the relevant transportable object or local object as needed.



A dialog titled 'Inactive Objects for DEVELOPER1' is open, listing transportable objects and their associated users

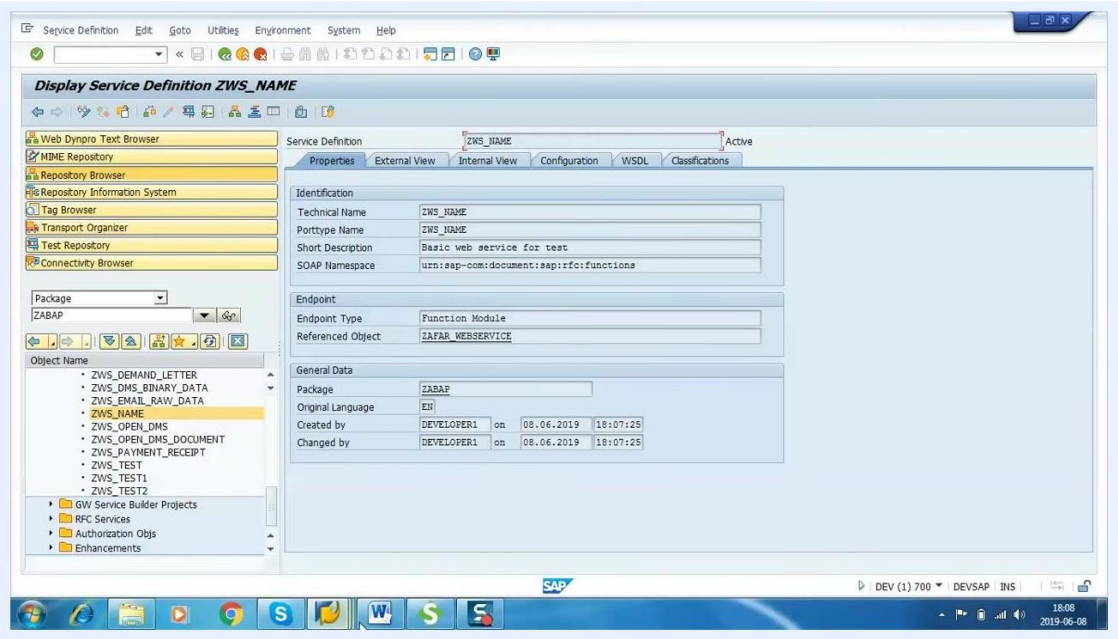
9. In the **Configuration** tab of the service definition, verify the security profile settings and ensure the authentication level matches your requirements (Low, Medium, High, or None). Confirm that the service definition (ZWS_NAME) has been activated by checking the status message at the bottom.



The Configuration tab of the Change Service Definition screen is open, showing the security profile, authentication settings, and activation status for ZWS_NAME

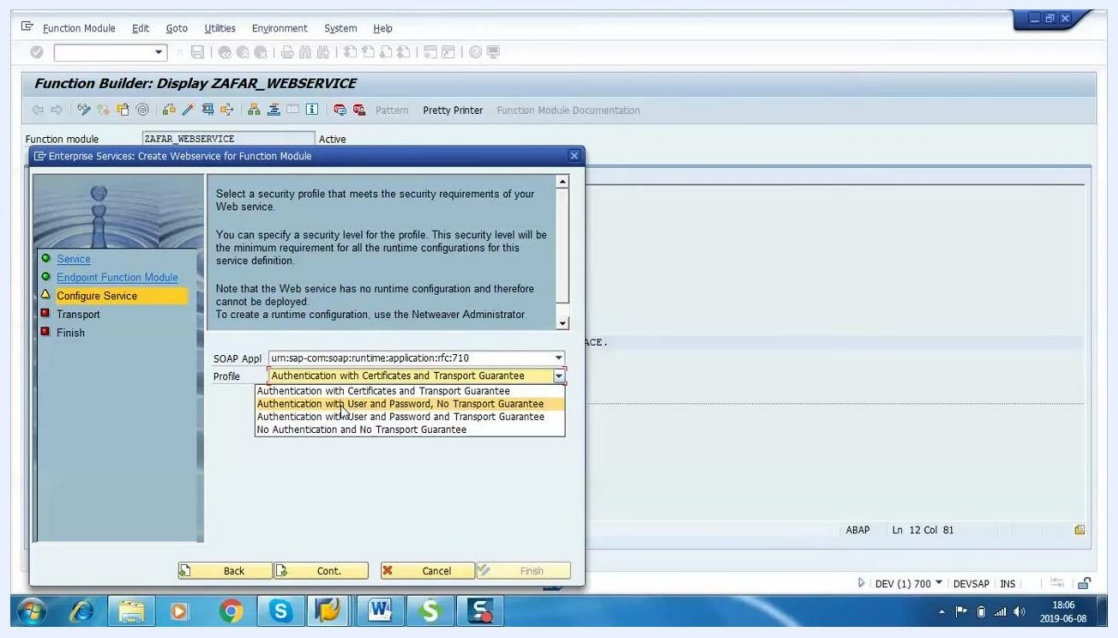
Step 5: Configure the web service in SOA Manager

1. In the ABAP Development Workbench (**SE80**), ensure your package (ZABAP) is selected in the Object Navigator. Expand the **Service Definitions** node to confirm that ZWS_NAME appears in the list of available service definitions.

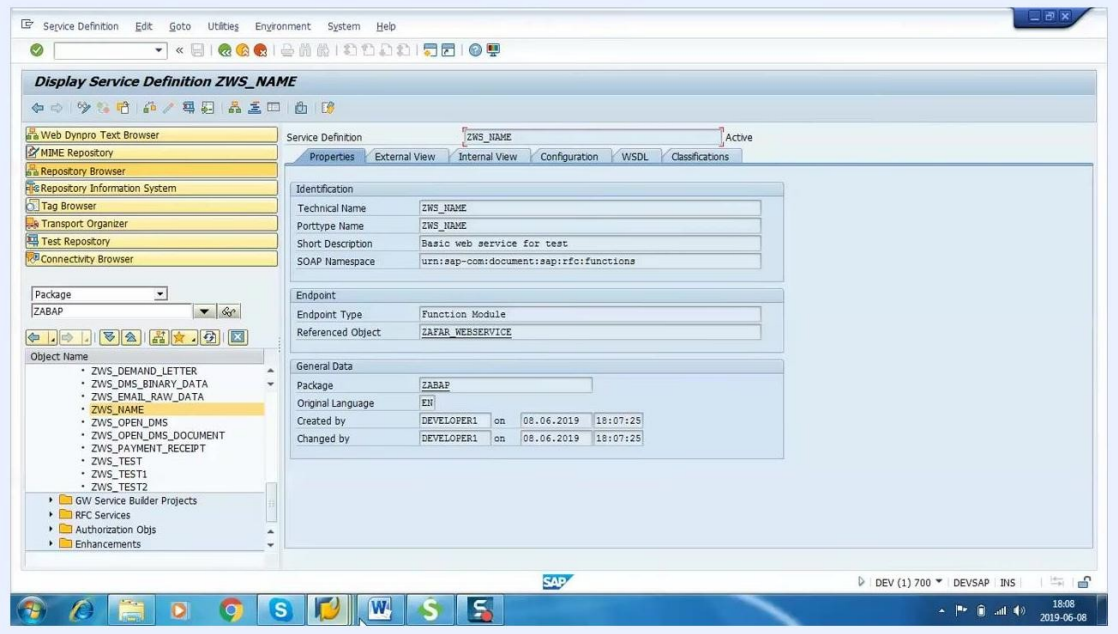


ABAP Development Workbench is open. The Object Navigator shows the ZABAP package expanded, with Service Definitions listed including ZWS_NAME. The main area displays 'ABAP DEVELOPMENT WORKBENCH' in orange text

2. Double-click on ZWS_NAME in the Service Definitions list. The **Display Service Definition ZWS_NAME** screen appears, showing detailed properties including Technical Name, Porttype Name, Short Description, SOAP Namespace, Endpoint Type, Referenced Object, Package, and Created/Changed by information. Available tabs include Properties, External View, Internal View, Configuration, WSDL, and Classifications.

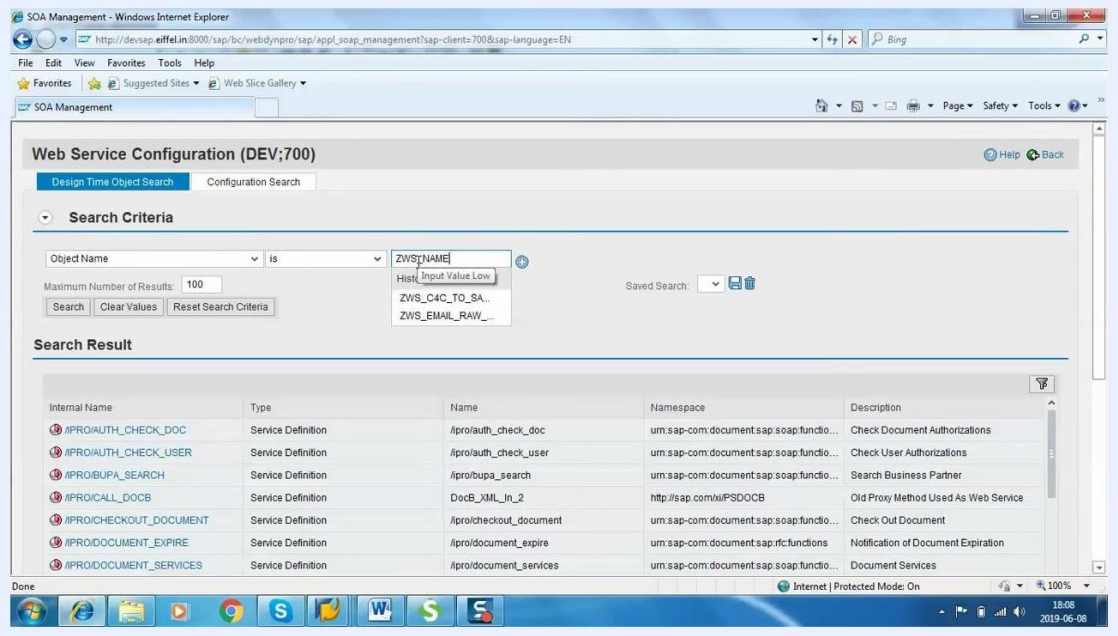


The Change Service Definition screen for ZWS_NAME is shown, displaying properties such as technical name, porttype name, description, SOAP namespace, endpoint type, referenced object, and package



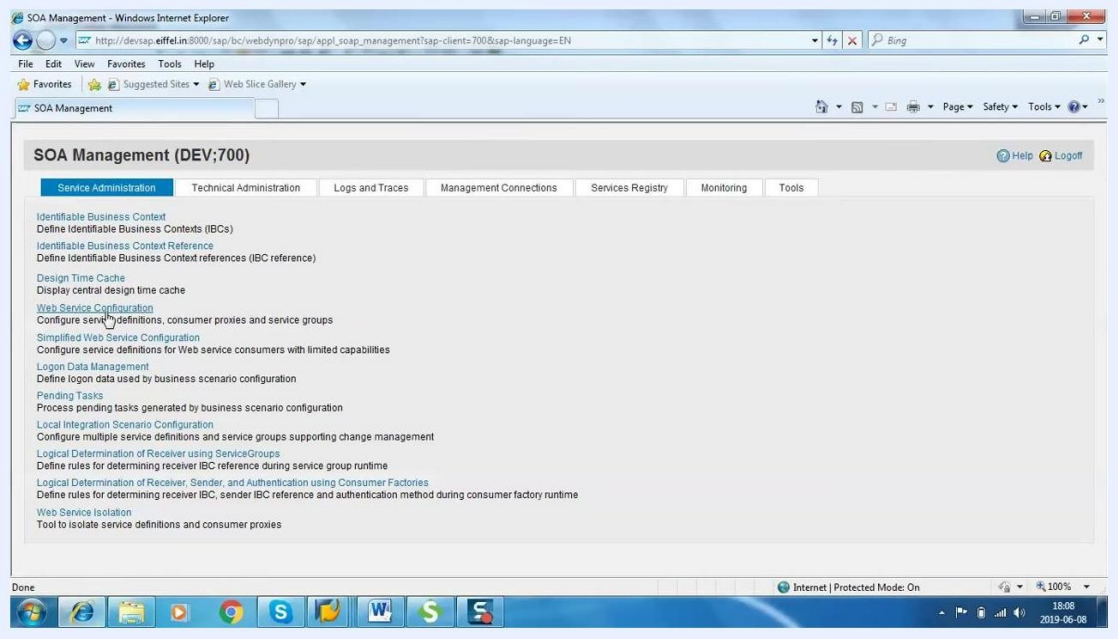
Display Service Definition ZWS_NAME screen is open, showing all service properties and metadata

3. Launch your web browser and navigate to the SOA Manager URL (for example, http://devsap.eiffel.in:8000/sap/bc/webdynpro/sap/appl_soap_management). If prompted, enter your SAP username and password to log in.



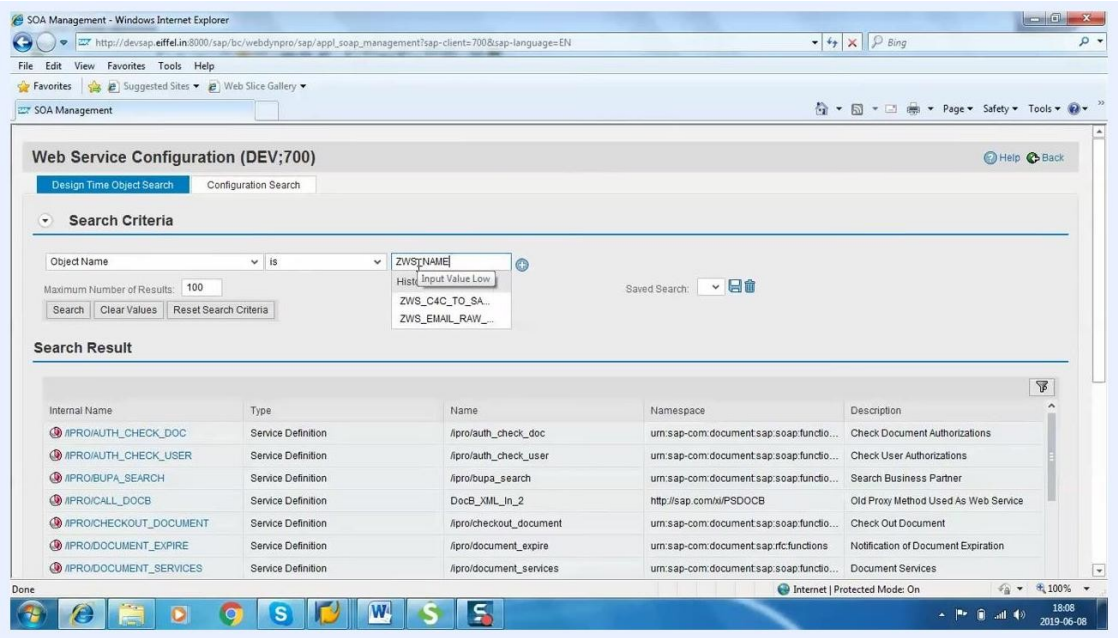
Internet Explorer is open with the SAP Easy Access screen in the background and the Internet Explorer 8 welcome/setup dialog in the foreground

4. In the SOA Management interface, ensure you are on the **Service Administration** tab. Locate and click on **Web Service Configuration** in the list of available options.



SOA Management screen is open in Internet Explorer. The Service Administration tab is selected, and 'Web Service Configuration' is visible in the list

5. In the **Web Service Configuration (DEV;700)** screen, under **Design Time Object Search**, set **Object Name** to is and enter ZWS_NAME in the field. Ensure **Maximum Number of Results** is set (for example, 100), then click **Search** to locate your service definition. The search results display below, listing matching service definitions.

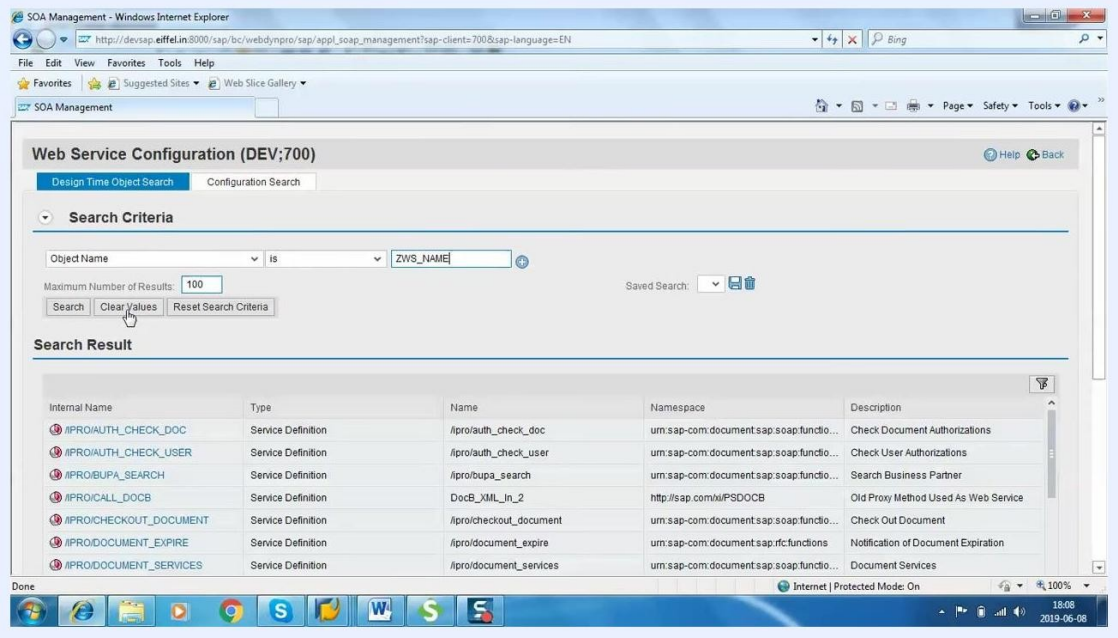


Web Service Configuration screen shows 'Object Name is ZWS_NAME' in the search criteria. The cursor is near the Search button

6. In the search results, select your service definition (ZWS_NAME) and click **Create Service** to start the binding configuration wizard. In the **Configuration of New Binding for Service Definition 'ZWS_NAME'** screen, fill in:

   - **Service Name:** ZWS_NAME
   - **Service Description Text:** web service for name

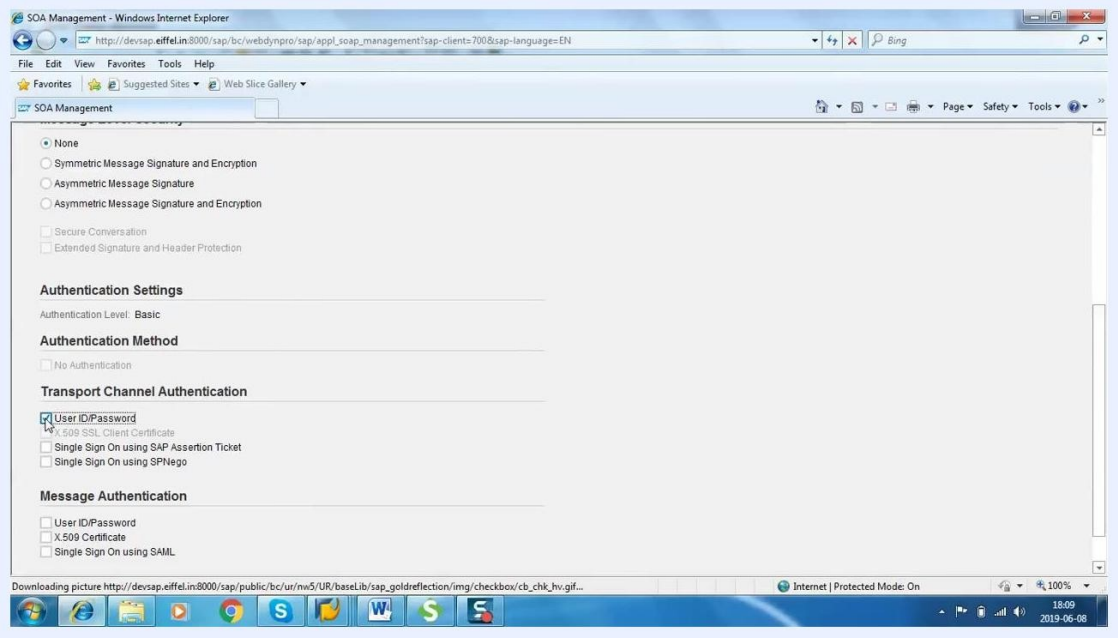
- **New Binding Name:** ZWS_NAME



Configuration of New Binding for Service Definition 'ZWS_NAME' is open. Service Name and New Binding Name fields are being filled with ZWS_NAME

7. Click **Next** to continue to the Provider Security step.

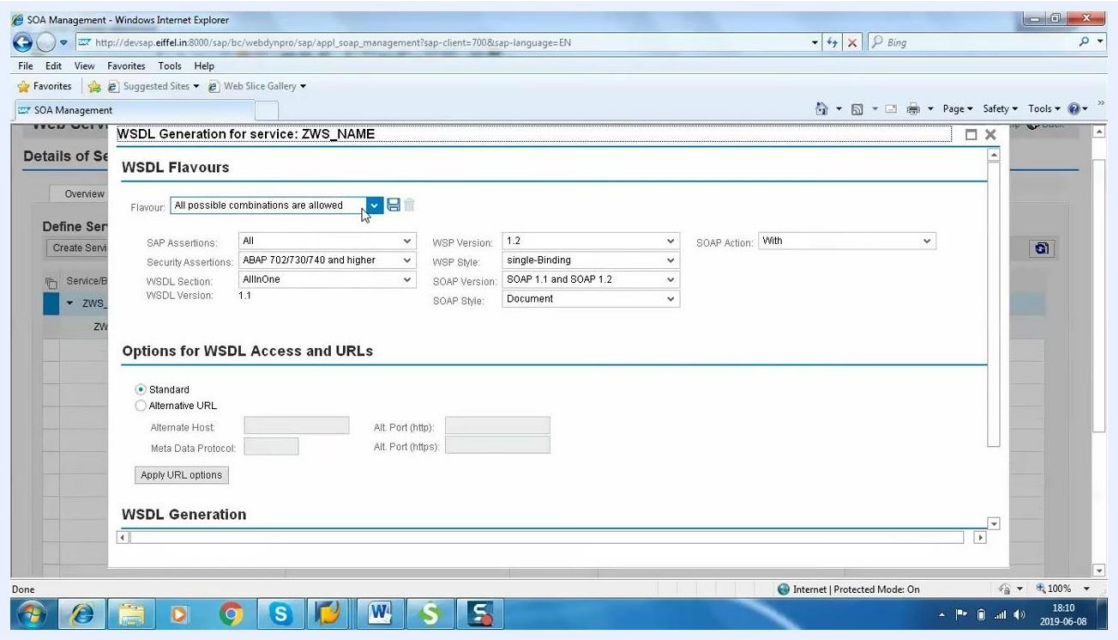
8. In the **Provider Security** step, locate the **Transport Channel Authentication** section and select the checkbox for **User ID/Password** to enable basic authentication for the web service. Leave other options (X.509 SSL Client Certificate, Single Sign On using SAP Assertion Ticket, Single Sign On using SPNego) unchecked unless required for your scenario. The **Authentication Level** is set to "Basic".



SOA Management screen showing Provider Security step. 'User ID/Password' is checked under Transport Channel Authentication. Other authentication options are visible but not selected

9. Click **Next** to advance to the **SOAP Protocol** step. Review the Transport Binding and Message Attachment Handling settings as needed.

10. After completing the wizard, you return to the **Details of Service Definition: ZWS_NAME** screen. Under the **Configurations** tab, verify that your service binding (ZWS_NAME) is listed with the state **Active** and the description "web service for name." The creation type is shown as "Manually created."



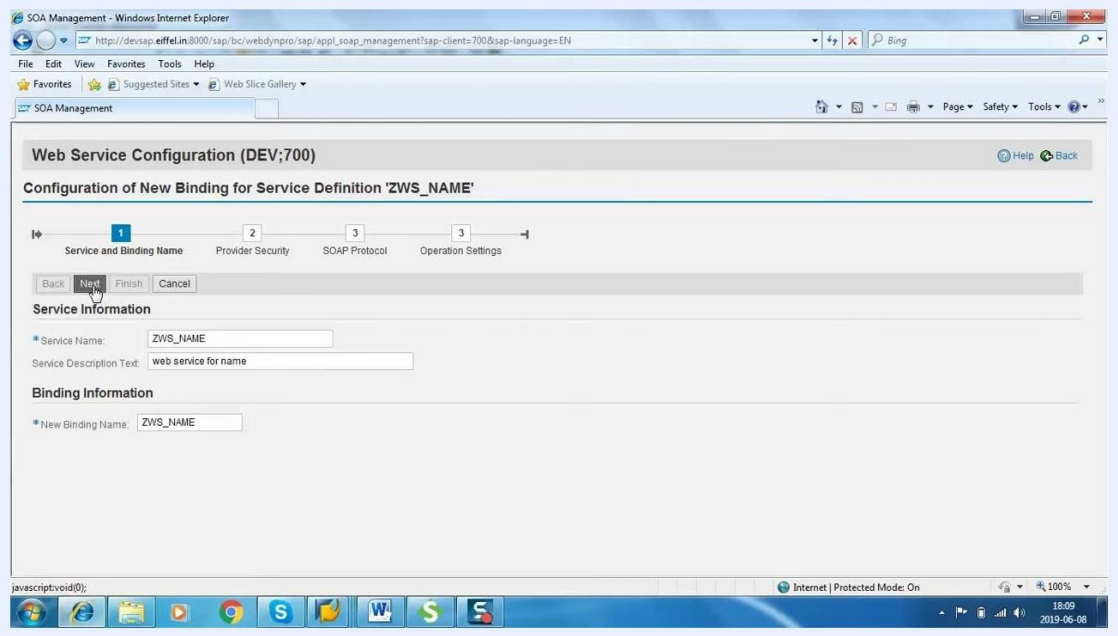
Details of Service Definition: ZWS_NAME screen. Service binding ZWS_NAME is listed as Active with description and creation type

Step 6: Export the WSDL file

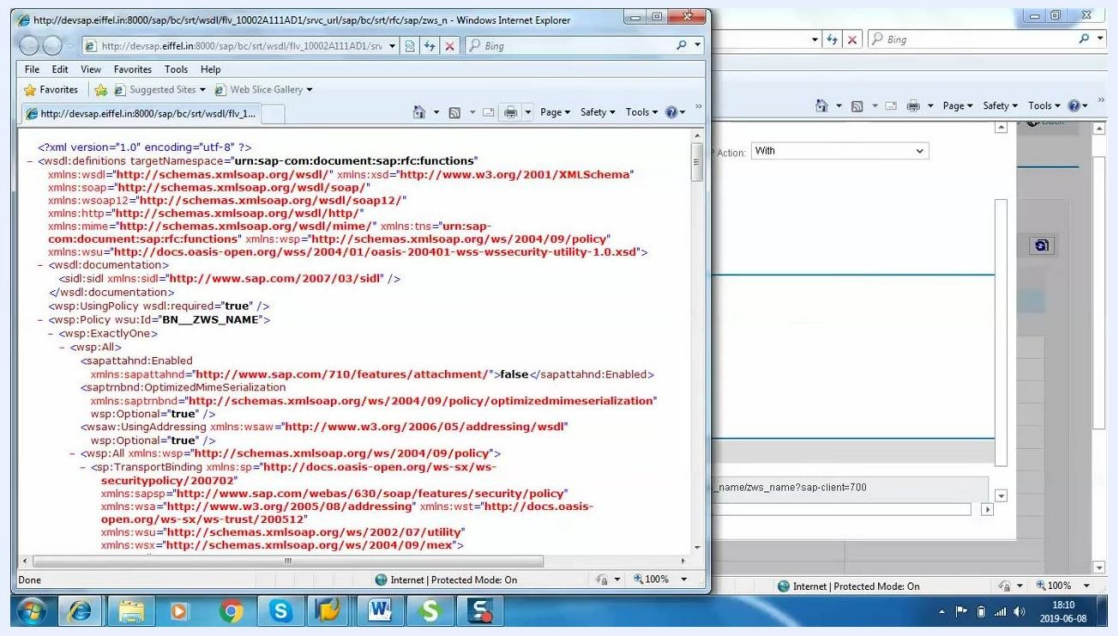
1. To use the web service outside of SAP (for example, in a non-SAP system), you need the WSDL file. In the **Configurations** tab, locate your active service binding and initiate WSDL generation by clicking a WSDL or export icon.
2. In the **WSDL Generation for service: ZWS_NAME** dialog, set the following options:

    - **Flavour:** All possible combinations are allowed
    - **SAP Assertions:** All
    - **Security Assertions:** ABAP 702/7.30/7.40 and higher
    - **WSDL Section:** AllInOne
    - **WSDL Version:** 1.1
    - **WSP Version:** 1.2
    - **WSP Style:** single-Binding

- **SOAP Version:** SOAP 1.1 and SOAP 1.2
- **SOAP Style:** Document
- **SOAP Action:** With



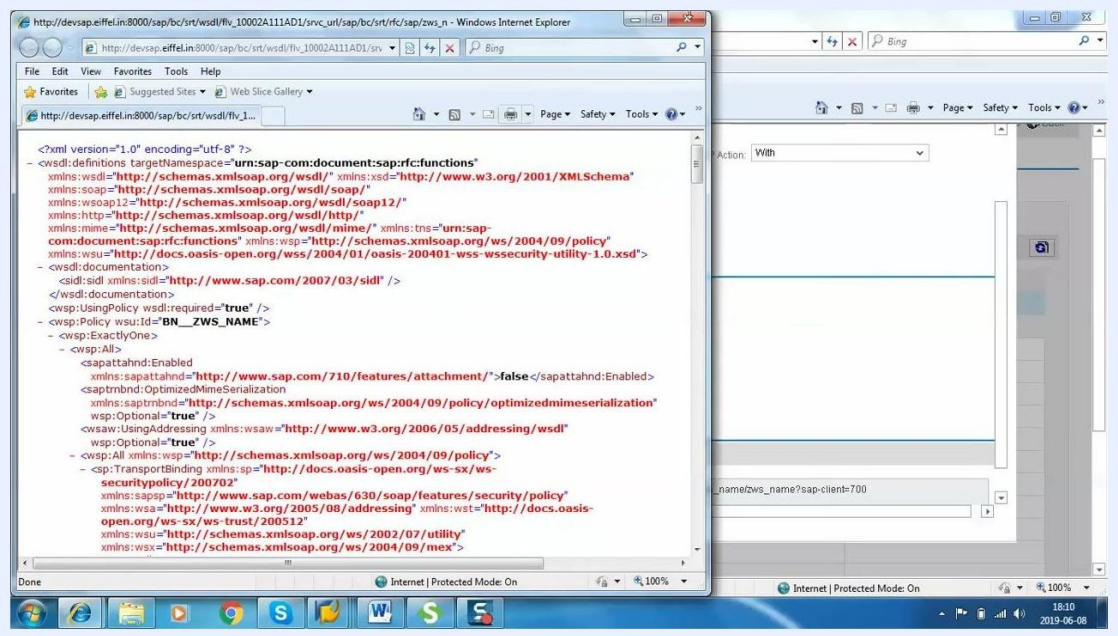
The Next button is being clicked in the binding configuration wizard, with all fields filled for ZWS_NAME



WSDL Generation dialog for service ZWS_NAME. All possible combinations are allowed for Flavour. Various dropdowns for assertions, versions, and styles are set as described

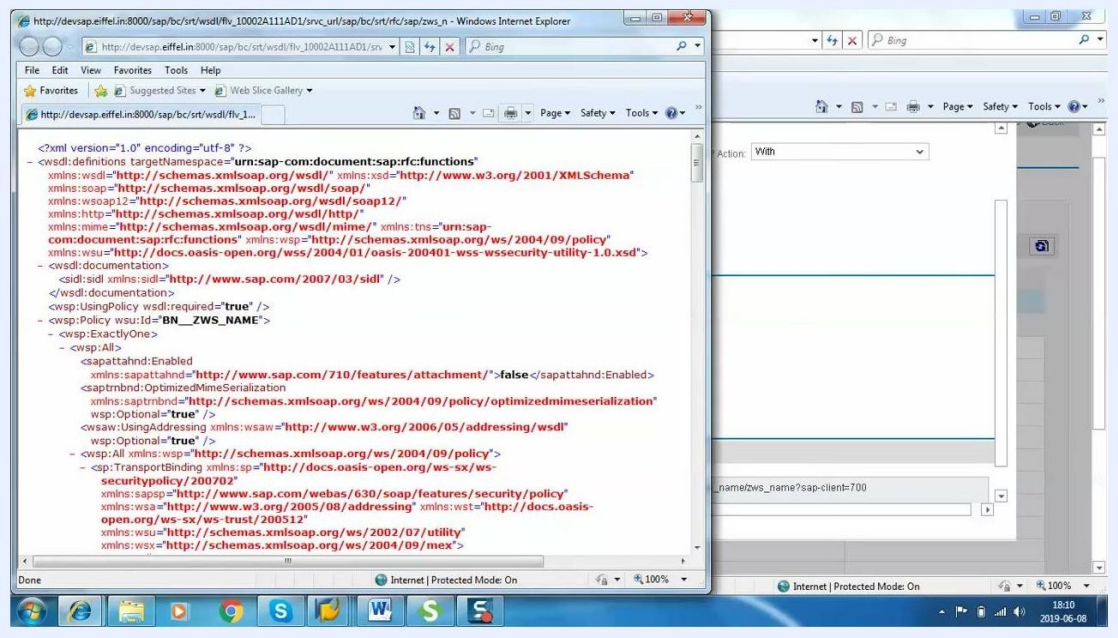
Under **Options for WSDL Access and URLs**, select **Standard**.

3. Click the **Save** (disk icon) button in the WSDL Generation dialog to save the selected WSDL flavour.



WSDL Generation dialog with the Save (disk) icon being clicked to save the WSDL flavour

4. The WSDL file opens in a new browser window or tab, displaying the XML content for the web service definition. Review the XML to ensure it contains the correct target namespace, service name, and policy details.

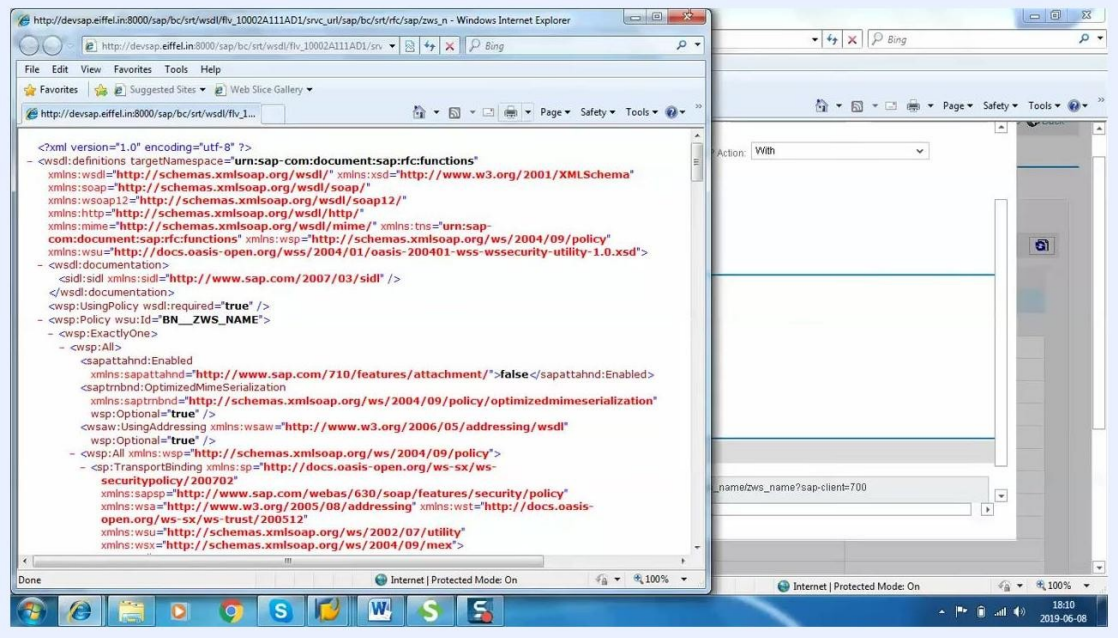


Browser window displaying the WSDL XML content for ZWS_NAME. The XML includes targetNamespace, definitions, and policy elements

5. In the browser window with the WSDL XML, go to the **File** menu and select **Save As**. In the **Save As** dialog:

    - Set the **File name** to zws_name.wsdl.
    - Ensure **Save as type** is set to "XML Files (*.xml)".
    - Choose the desired location (for example, Desktop or a project folder).

- Click **Save** to store the WSDL file locally.

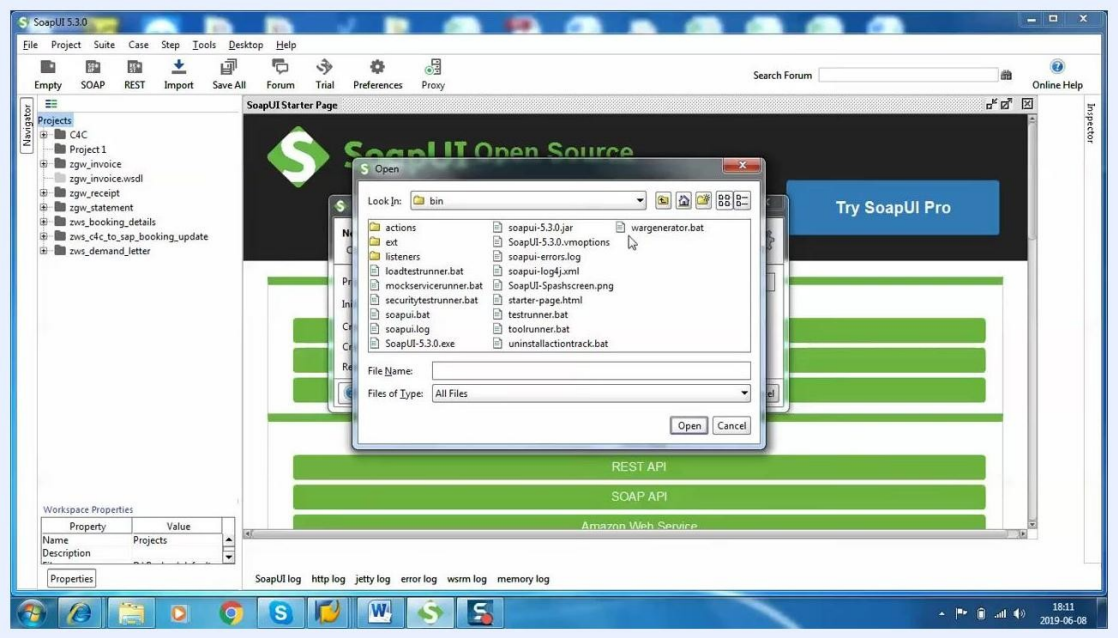


Save As dialog open with file name set to zws_name.wsdl and Save as type set to XML Files. The Save button is being clicked

6. Navigate to your desktop or the folder where you saved the file. Look for zws_name.wsdl and confirm that it appears with the correct name and .wsdl extension.
7. Optionally, double-click the zws_name.wsdl file to open it in your default XML or text editor and confirm that it opens without errors and displays the expected WSDL XML content.

Step 7: Test the web service using SoapUI

1. On your Windows desktop, find the "SoapUI 5.3.0" shortcut icon (green S logo) and double-click it to open SoapUI.



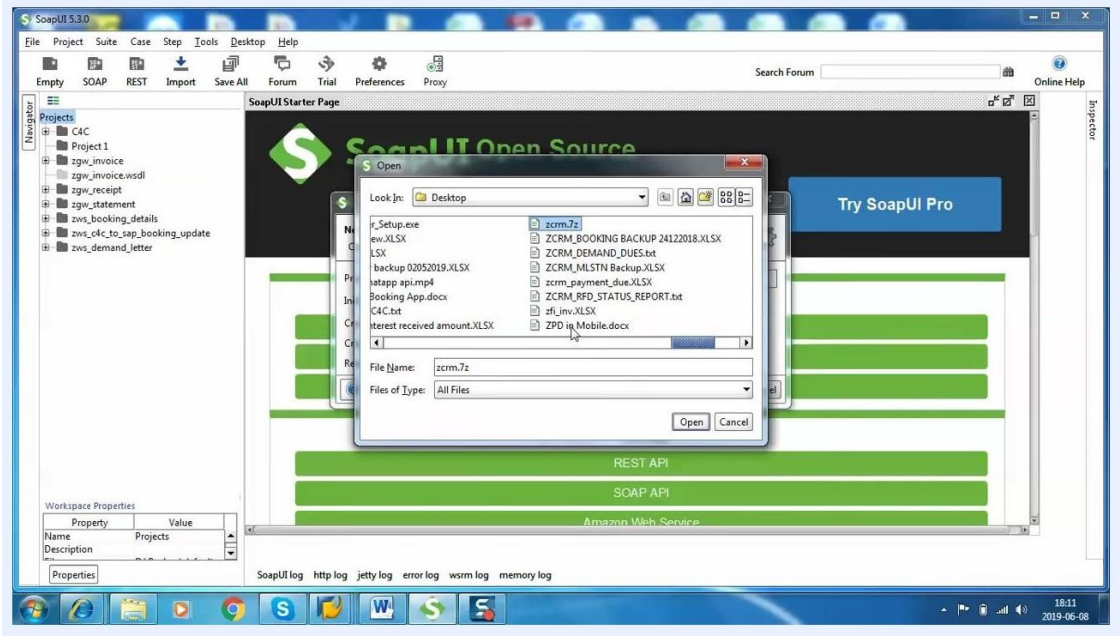
Windows desktop with many files and the SoapUI 5.3.0 shortcut icon highlighted, ready to be launched

2. In the SoapUI main window, click the **File** menu in the top left and select **New SOAP Project** from the dropdown menu.

3. In the **New SOAP Project** dialog, click **Browse** next to **Initial WSDL**. In the file dialog, navigate to the location of your WSDL file — if your file is on the desktop, select **Desktop** from the left panel. Find and select your WSDL file (zws_name.wsdl), then click **Open** to confirm your selection.



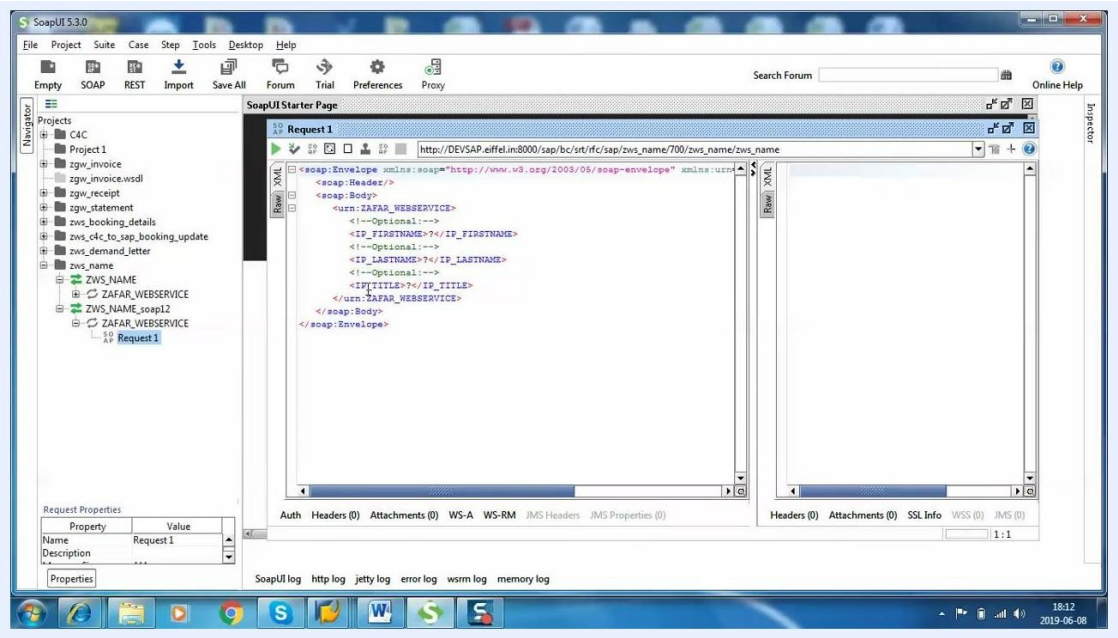
Windows desktop with many files and shortcuts visible. The file 'zws_name.wsdl' is present among other documents and icons



SoapUI open file dialog showing navigation to select a WSDL file from the desktop

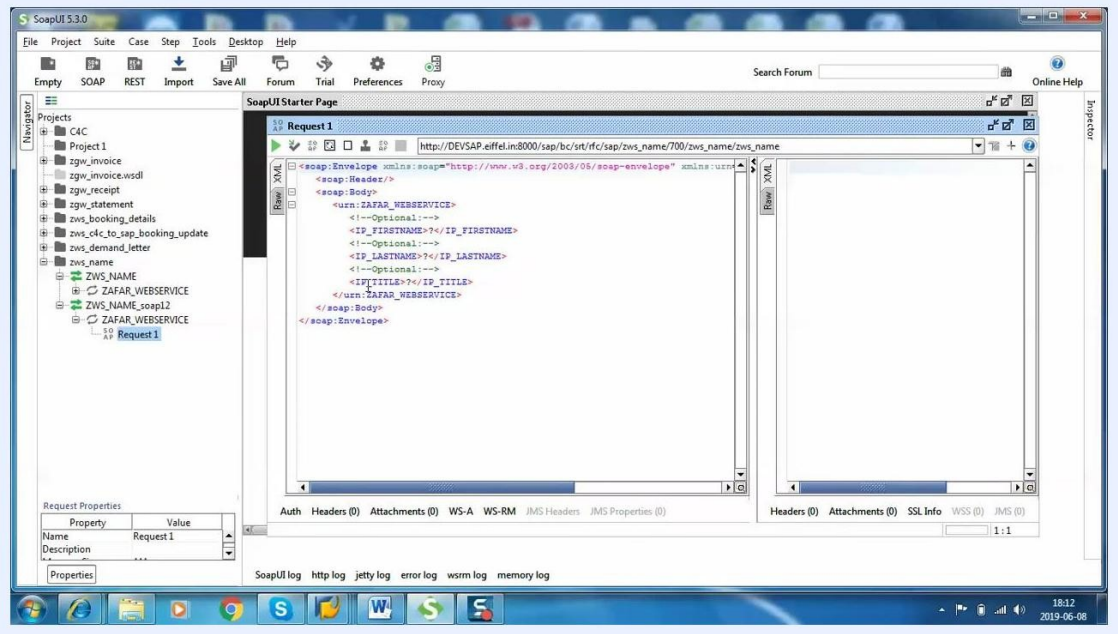
4. In the **New SOAP Project** dialog:

    - Set **Project Name** to zws_name.
    - Ensure **Initial WSDL** points to the correct file path (for example, D:\Backup\Desktop\zws_name.wsdl).
    - Check **Create sample requests for all operations?** if you want SoapUI to generate sample requests.
    - Optionally, check **Create TestSuite** if you want a test suite created.
    - Click **OK** to create the project.



SoapUI New SOAP Project dialog with project name and WSDL file path filled in

5. In the left **Projects** panel, expand the new project node (zws_name), then expand the interface and operation nodes until you see "Request 1" under your web service (ZAFAR_WEBSERVICE). Double-click **Request 1** to open the request editor.

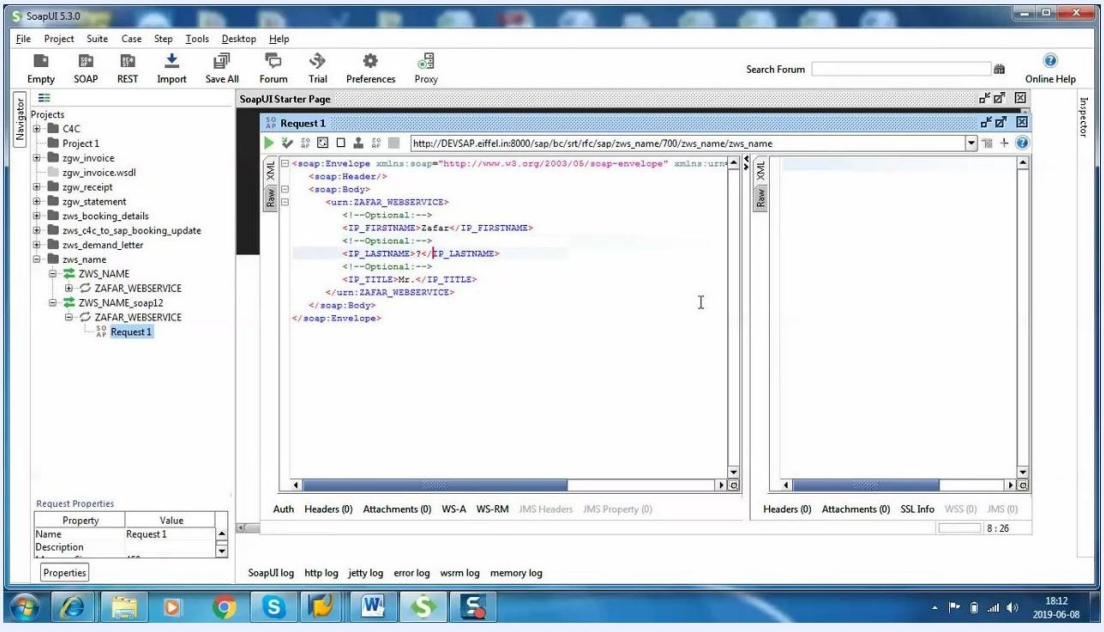


SoapUI project navigator showing the expanded project tree with Request 1 visible

6. In the request editor, locate the XML body and fill in the required fields:

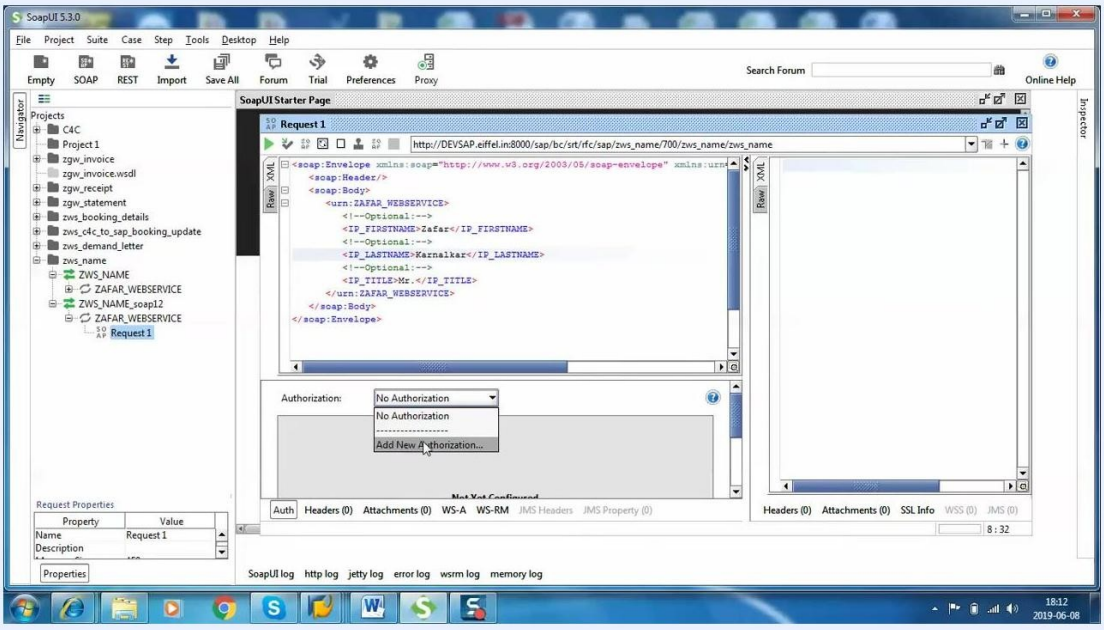
   - <IP_FIRSTNAME>: Enter the first name (for example, Zafar).
   - <IP_LASTNAME>: Enter the last name (for example, Karnalkar).

- <IP_TITLE>: Enter the title (for example, Mr.).



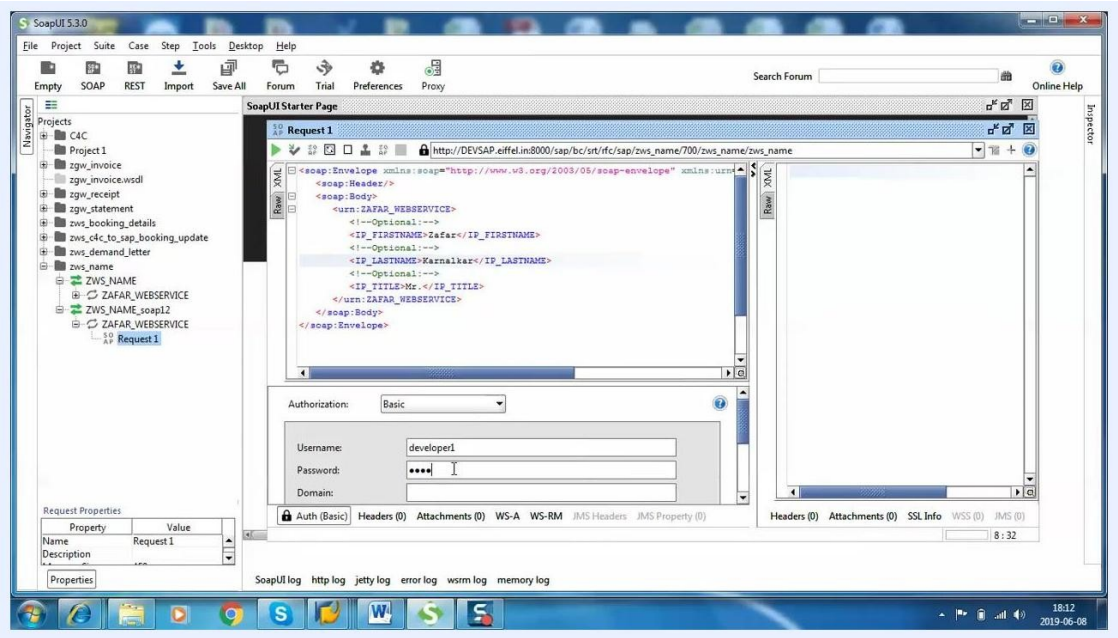
SoapUI request editor with XML body fields for first name, last name, and title filled in

7. At the bottom of the request editor, click the **Auth** tab. In the **Authorization** dropdown, select **Add New Authorization...**, choose **Basic** as the authorization type, and click **OK** to confirm. Enter your SAP user ID and password in the provided fields.



SoapUI request editor with the Auth tab open and Add New Authorization selected

8. In the SoapUI request editor, locate the icons above the XML request body. The **Submit** icon (green arrow) sends the SOAP request to the web service, while a **Test** icon may also be available for running tests or assertions. Hover over each icon to see tooltips describing its function, then click **Submit** (green arrow) to send your request.

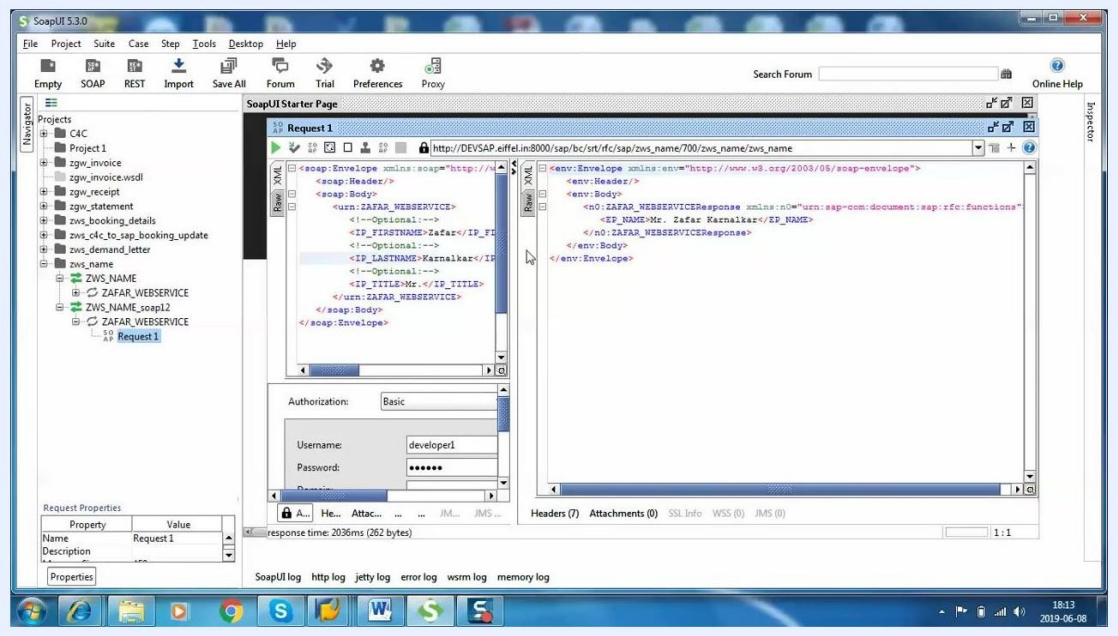


SoapUI request editor showing the XML request, authentication fields, and the toolbar with the submit (green arrow) icon highlighted above the request body

9. After clicking **Submit**, observe the right pane of the request editor. The response XML is displayed, showing the output from the web service. In this example, the response contains:

```
<EP_NAME>Mr. Zafar Karnalkar</EP_NAME>
```

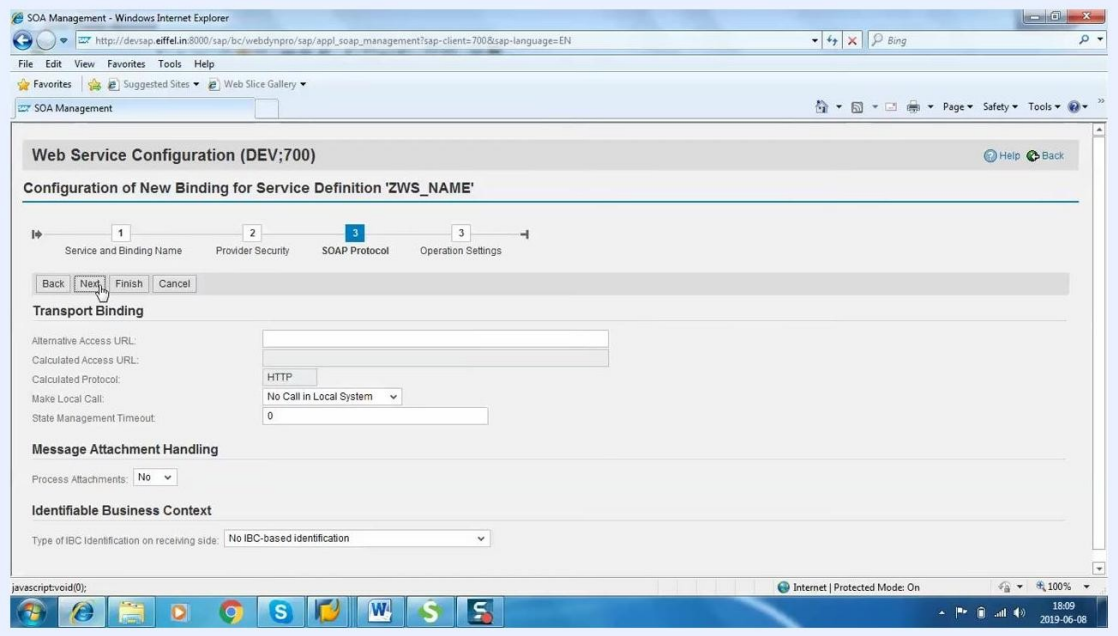
Confirm that the response matches the expected output for your input parameters.



SoapUI request editor with the response pane showing the returned XML containing EP_NAME with the value "Mr. Zafar Karnalkar"

What's next

With your web service created, configured, and verified in SoapUI, you now have a working SOAP web service built on a function module that external systems can consume. You can reuse this same workflow — creating a function group, building a function module, generating a web service definition, configuring the binding in SOAMANAGER, and exporting the WSDL — to expose additional ABAP logic as web services for other integration scenarios.



SOAP Protocol step of the binding wizard is open. The Next button is being clicked. Fields for Transport Binding and Message Attachment Handling are visible