**Technical Specification Form**

**Product Line Custom Table CUD Operation**

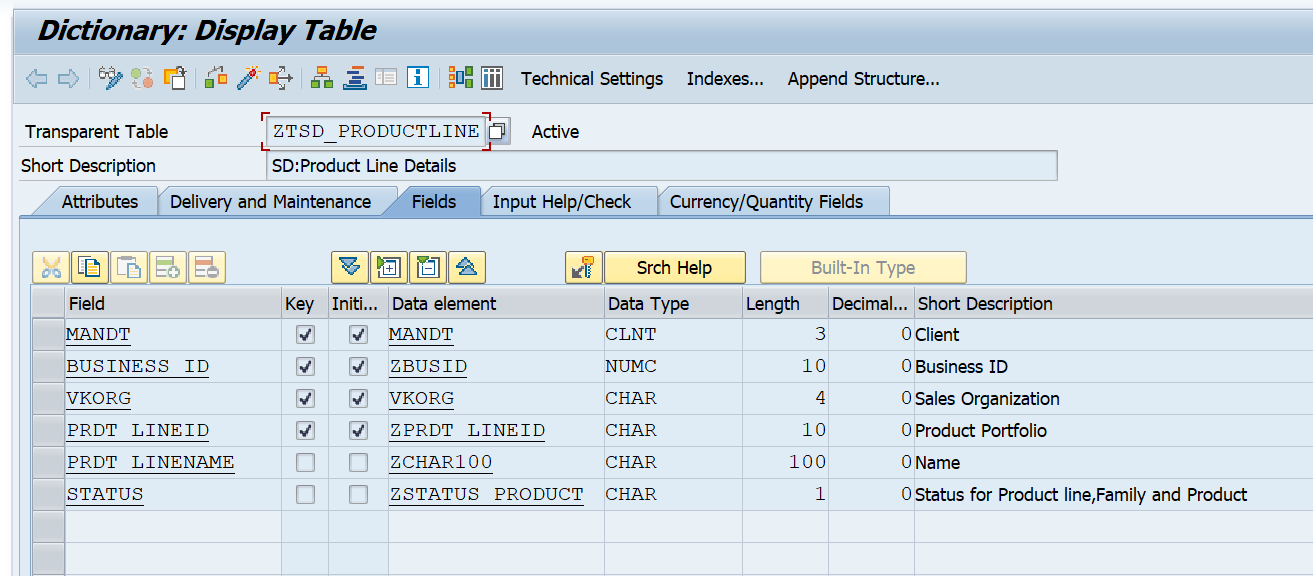
Created by: 9000RUPESH (AIRDIT Software)

**Technical Specification Document Content:**

1. Business Requirement.
2. Development Object used.
3. Development Approach step by step.
4. Testing API with Payload and Testing screen’s Screenshot.
5. **Business Requirement.**

Here business user wants an OData API through which he/she can create/update/Delete mass record in a SAP Custom table (ZTSD\_PRODUCTLINE).

This following is the custom Table detail:



**2.Development Object used.**

For development I used the following sap Tools/tcode:

* **DDIC Structure**

Here DDIC Structure used for Crete an Entity in OData Project.

Tcode:SE11

* **OData Project**

Creating an OData (Open Data Protocol) project is essential for building and consuming standardized, RESTful APIs that allow applications to exchange data seamlessly over the web. The project defines the data model and the interface for performing create, read, update, and delete (CRUD) operations on server-side data.

Tcode:SEGW

* **OData service Registration**

OData service registration in SAP is done to make the created OData service available for consumption by other applications, such as SAP Fiori or mobile apps. This step formally activates the service, enabling external systems to connect and access the data exposed through the OData API. Without registration, the service remains just a design-time object and is not accessible for runtime use.

Tcode:/N/IWFND/MAINT\_SERVICE

* **OData API Testing**

Here for testing the OData API’s , we have to use the tcode /N/IWFND/MAINT\_SERVICE or /N/IWFND/GW\_CLIENTG in order to test whether it’s working as per our expectation or not.

**3.Development Approach step by step.**

Before start the development Step, please read the Note section which will clear the confusion while going through the development Steps.

**Note:**

In a standard OData implementation, each API call is designed to handle only a single record per Create, Update, or Delete (CUD) operation. However, in this development, there is a requirement to process multiple records (mass operations) within a single API request.

To achieve this, a customized approach has been implemented using the POST operation of the OData service. Two additional fields have been introduced in the entity structure:

1. DataString – This field carries the mass data in a stringified JSON format.
2. Operation – This field specifies the type of operation to be performed (CR for Create, UR for Update, DR for Set Status as X/Delete, and UD for Remove the Status from X/undo Delete).

The entire payload is sent through a single POST API call. On the backend, within the CREATE\_ENTITY method, the following logic is executed:

The DataString field is deserialized to extract the actual data records.

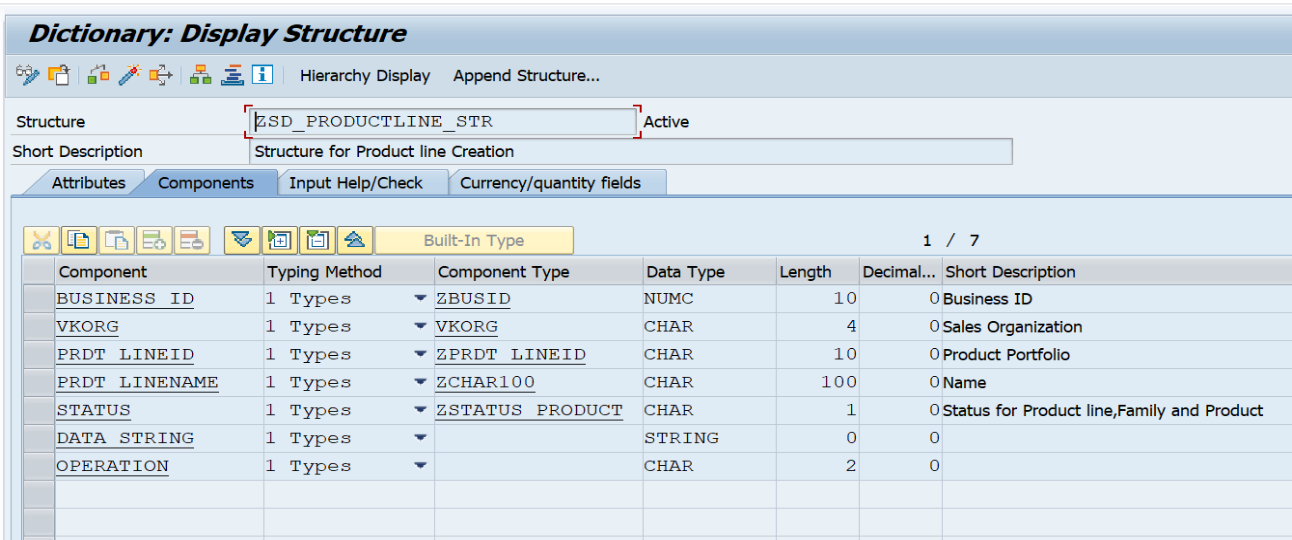
Based on the Operation code, the corresponding logic for Create, Update, or Delete is triggered.

This approach eliminates the need to redefine or separately implement the UPDATE\_ENTITY and DELETE\_ENTITY methods.

All operations are handled efficiently through a single POST API, allowing flexibility for mass data processing while maintaining OData service structure and simplicity.

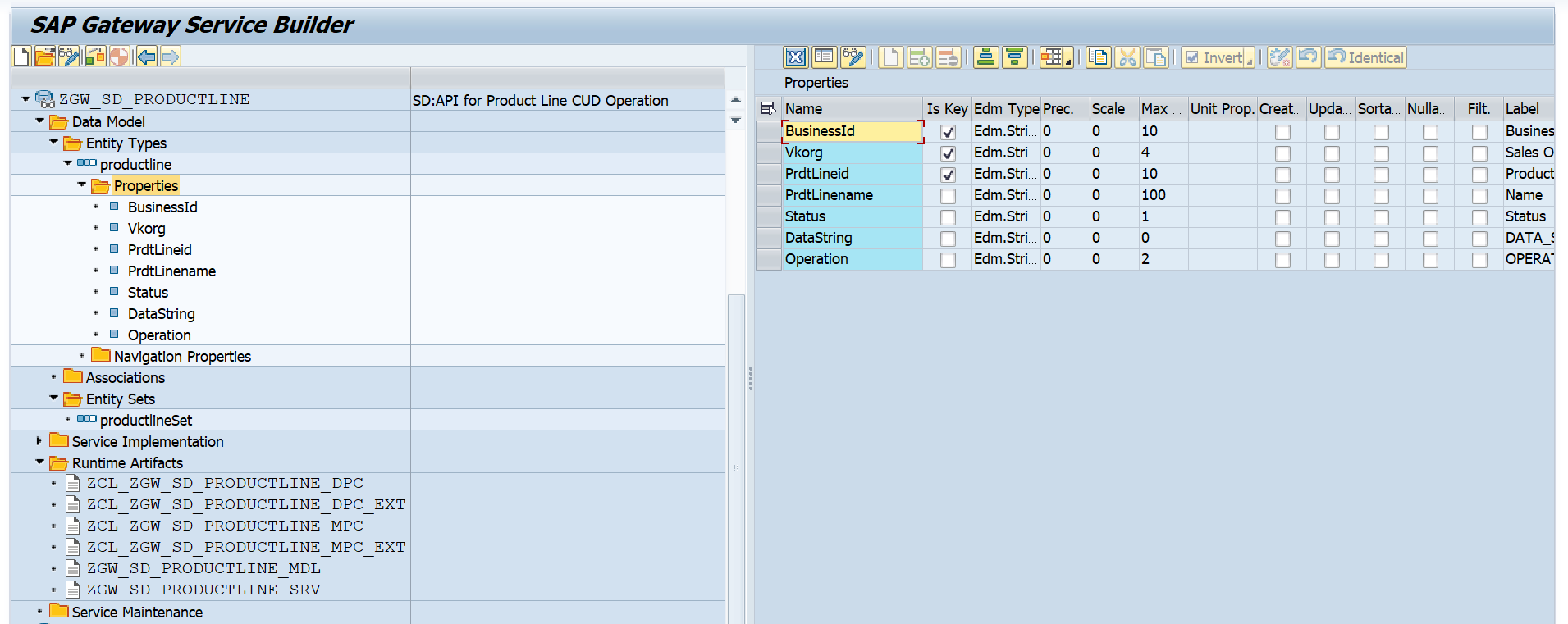
**Step 1: Crete a DDIC Structure**

This DDIC Structure will contain all the fields available in the database table along with two additional fields mentioned in the Note.

Name of DDIC Structure: ZSD\_PRODUCTLINE\_STR

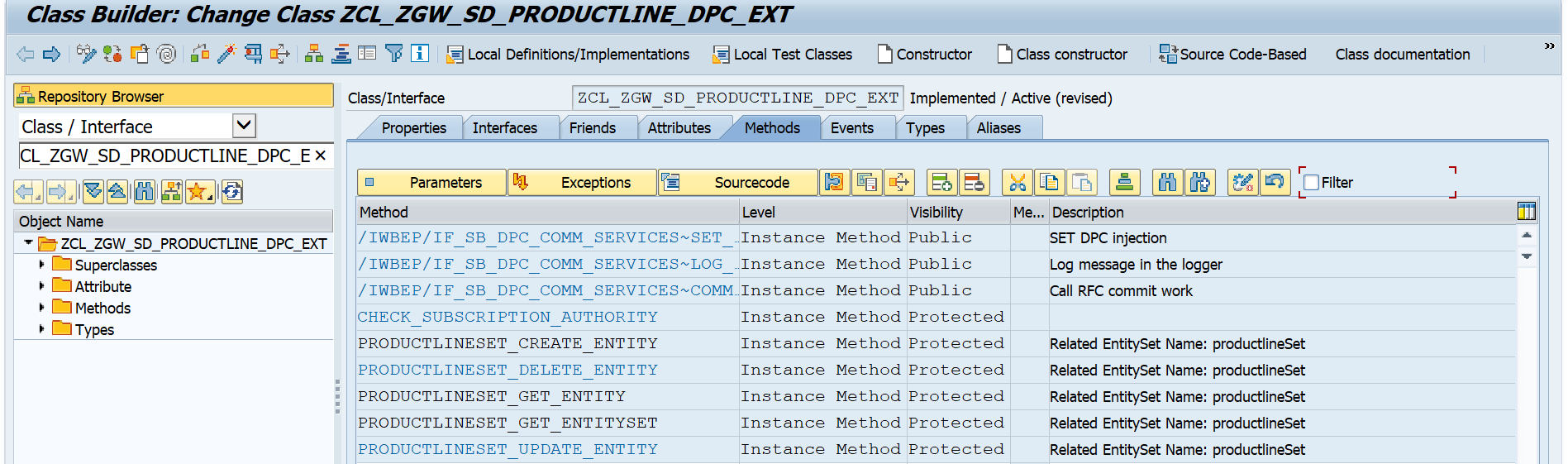
**Step 2: Create and SEGW OData project.**

Here for creating the ODATA project I used the normal approach and while creating the Entity I used **import from DDIC Structure, then** Regenerate the OData project for Future Process.

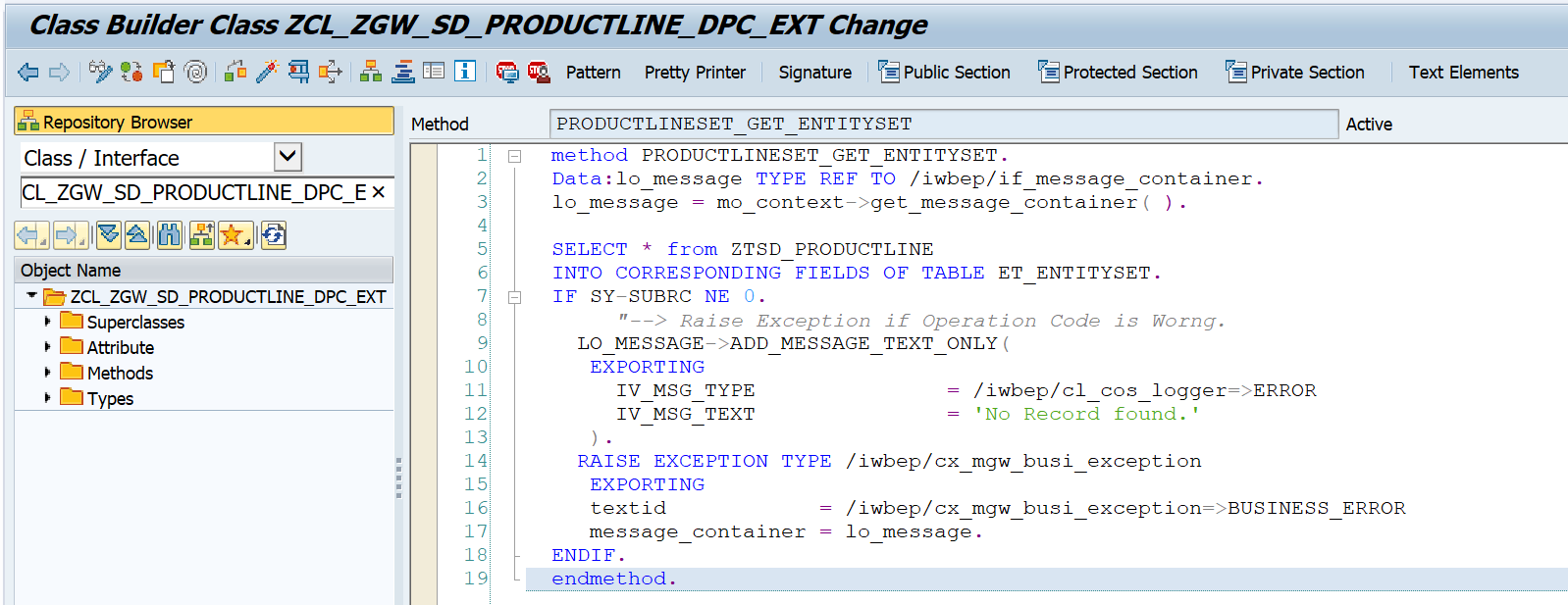
* Project name: **ZGW\_SD\_PRODUCTLINE**
* Entity Type name: **productline**
* Entity Set Name: **productlineSet**
* OData Service Name: **ZGW\_SD\_PRODUCTLINE\_SRV**

**Step 3: Redefine the Required method in DPC\_EXT Class for Business logic implementation.**

Here I redefine the following Methods:



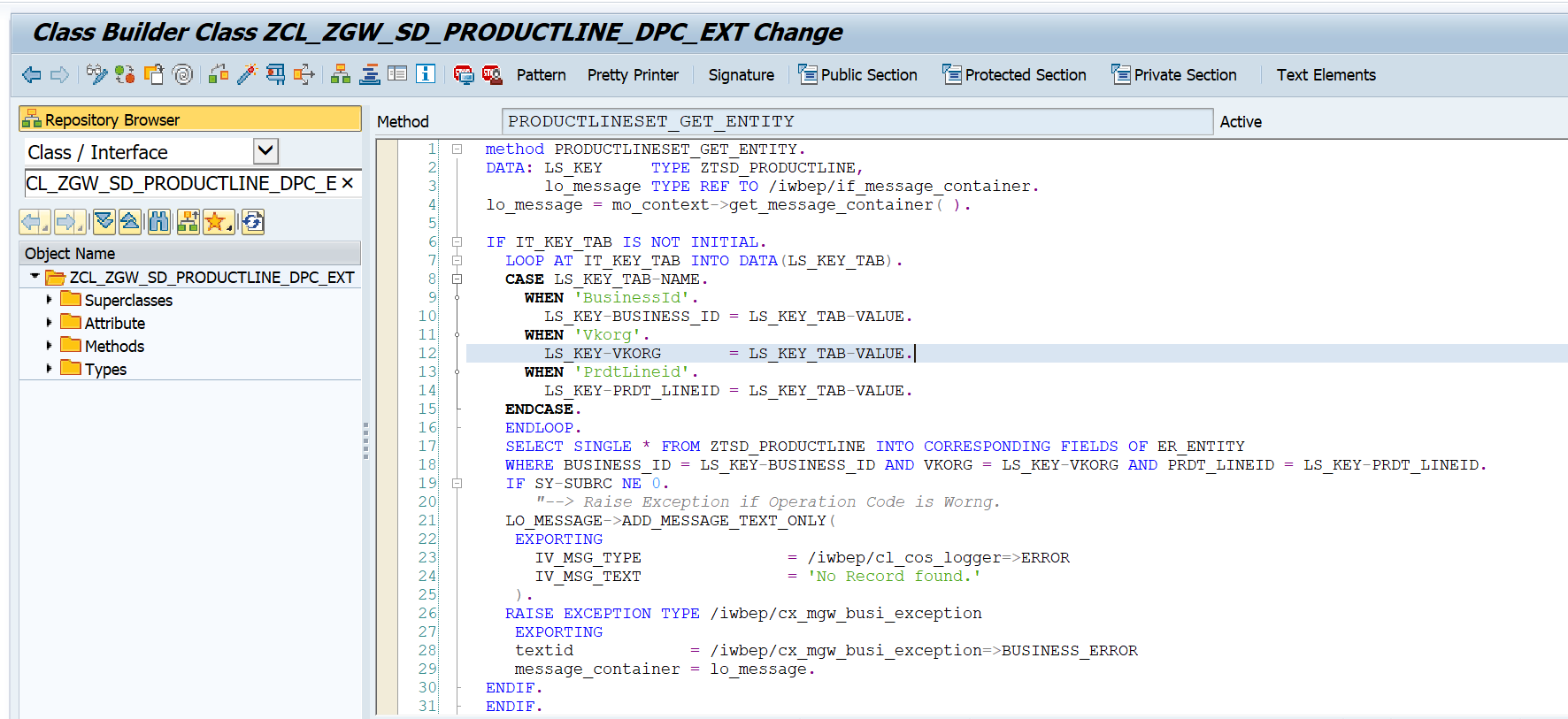
**3.1: Implement the “PRODUCTLINESET\_GET\_ENTITYSET”.**

Note: It will help us for Fetch CSRF Token.

**Business Logic for Copy:**

* method PRODUCTLINESET\_GET\_ENTITYSET.  
    Data:lo\_message TYPE REF TO /iwbep/if\_message\_container.  
    lo\_message = mo\_context->get\_message\_container( ).  
    
    SELECT \* from ZTSD\_PRODUCTLINE  
    INTO CORRESPONDING FIELDS OF TABLE ET\_ENTITYSET.  
    IF SY-SUBRC NE 0.  
         *"--> Raise Exception if Operation Code is Worng.*  
      LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
       EXPORTING  
         IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>ERROR  
         IV\_MSG\_TEXT               = 'No Record found.'  
       ).  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
       EXPORTING  
       textid            = /iwbep/cx\_mgw\_busi\_exception=>BUSINESS\_ERROR  
       message\_container = lo\_message.  
    ENDIF.  
    endmethod.

**3.2: Implement of “PRODUCTLINESET\_GET\_ENTITY”.**

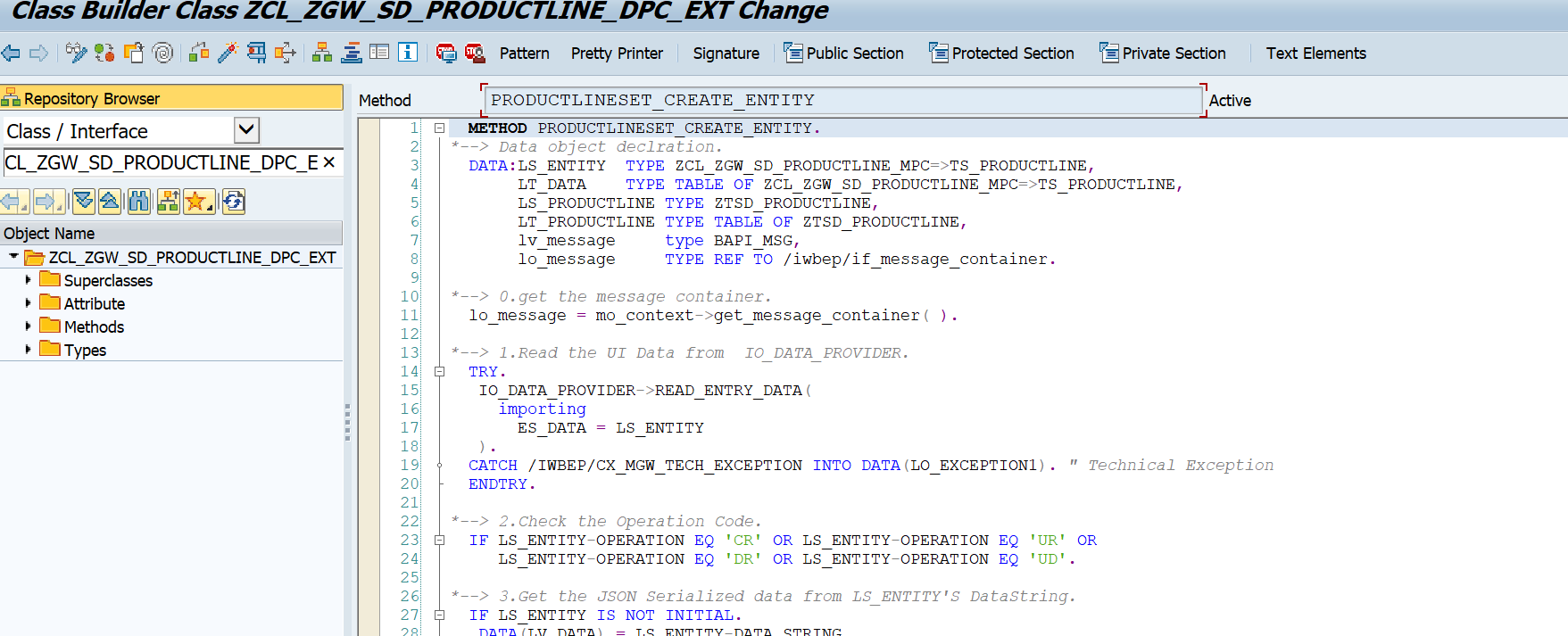


**Business Logic for Copy:**

* method PRODUCTLINESET\_GET\_ENTITY.  
    DATA: LS\_KEY     TYPE ZTSD\_PRODUCTLINE,  
          lo\_message TYPE REF TO /iwbep/if\_message\_container.  
    lo\_message = mo\_context->get\_message\_container( ).  
    
    IF IT\_KEY\_TAB IS NOT INITIAL.  
      LOOP AT IT\_KEY\_TAB INTO DATA(LS\_KEY\_TAB).  
      CASE LS\_KEY\_TAB-NAME.  
        WHEN 'BusinessId'.  
          LS\_KEY-BUSINESS\_ID = LS\_KEY\_TAB-VALUE.  
        WHEN 'Vkorg'.  
          LS\_KEY-VKORG       = LS\_KEY\_TAB-VALUE.  
        WHEN 'PrdtLineid'.  
          LS\_KEY-PRDT\_LINEID = LS\_KEY\_TAB-VALUE.  
      ENDCASE.  
      ENDLOOP.  
      SELECT SINGLE \* FROM ZTSD\_PRODUCTLINE INTO CORRESPONDING FIELDS OF ER\_ENTITY  
      WHERE BUSINESS\_ID = LS\_KEY-BUSINESS\_ID AND VKORG = LS\_KEY-VKORG AND PRDT\_LINEID = LS\_KEY-PRDT\_LINEID.  
      IF SY-SUBRC NE 0.  
         *"--> Raise Exception if Operation Code is Worng.*  
      LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
       EXPORTING  
         IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>ERROR  
         IV\_MSG\_TEXT               = 'No Record found.'  
       ).  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
       EXPORTING  
       textid            = /iwbep/cx\_mgw\_busi\_exception=>BUSINESS\_ERROR  
       message\_container = lo\_message.  
    ENDIF.  
    ENDIF.  
    endmethod.

3.3: Implement of “PRODUCTLINESET\_CREATE\_ENTITY”.

This single method is going to Handel Create, Update & Delete.

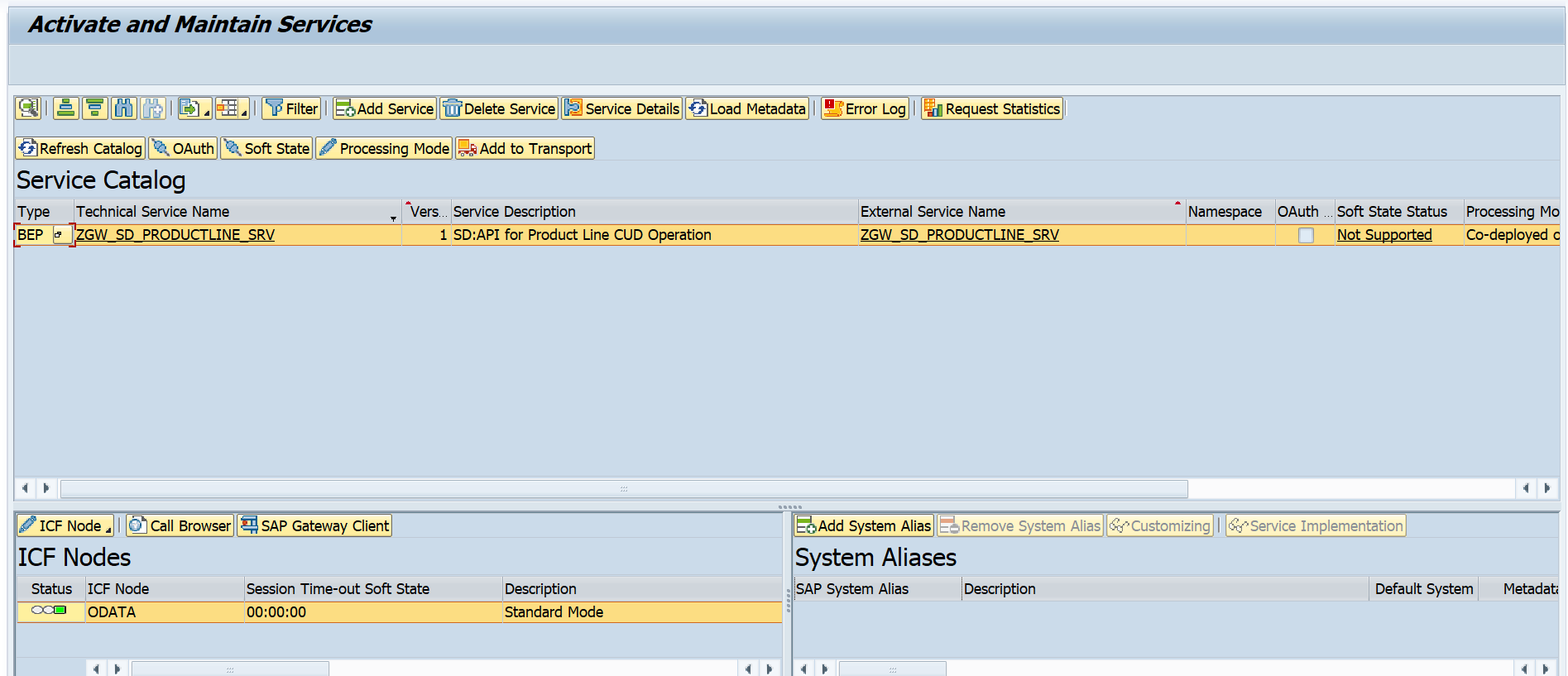


**Business Logic for Copy:**

  METHOD PRODUCTLINESET\_CREATE\_ENTITY.  
*\*--> Data object declration.*  
  DATA:LS\_ENTITY  TYPE ZCL\_ZGW\_SD\_PRODUCTLINE\_MPC=>TS\_PRODUCTLINE,  
       LT\_DATA    TYPE TABLE OF ZCL\_ZGW\_SD\_PRODUCTLINE\_MPC=>TS\_PRODUCTLINE,  
       LS\_PRODUCTLINE TYPE ZTSD\_PRODUCTLINE,  
       LT\_PRODUCTLINE TYPE TABLE OF ZTSD\_PRODUCTLINE,  
       lv\_message     type BAPI\_MSG,  
       lo\_message     TYPE REF TO /iwbep/if\_message\_container.  
  
*\*--> 0.get the message container.*  
  lo\_message = mo\_context->get\_message\_container( ).  
  
*\*--> 1.Read the UI Data from  IO\_DATA\_PROVIDER.*  
  TRY.  
   IO\_DATA\_PROVIDER->READ\_ENTRY\_DATA(  
     importing  
       ES\_DATA = LS\_ENTITY  
   ).  
  CATCH /IWBEP/CX\_MGW\_TECH\_EXCEPTION INTO DATA(LO\_EXCEPTION1). *" Technical Exception*  
  ENDTRY.  
  
*\*--> 2.Check the Operation Code.*  
  IF LS\_ENTITY-OPERATION EQ 'CR' OR LS\_ENTITY-OPERATION EQ 'UR' OR  
     LS\_ENTITY-OPERATION EQ 'DR' OR LS\_ENTITY-OPERATION EQ 'UD'.  
  
*\*--> 3.Get the JSON Serialized data from LS\_ENTITY'S DataString.*  
  IF LS\_ENTITY IS NOT INITIAL.  
   DATA(LV\_DATA) = LS\_ENTITY-DATA\_STRING.  
   CHECK LV\_DATA IS NOT INITIAL.  
   /UI2/CL\_JSON=>DESERIALIZE(  
     exporting  
       JSON             = LV\_DATA                                        *" JSON string*  
       PRETTY\_NAME      = /UI2/CL\_JSON=>PRETTY\_MODE-CAMEL\_CASE           *" Pretty Print property names*  
     changing  
       DATA             = LT\_DATA                                        *" Data to serialize*  
   ).  
  ENDIF.  
  
  *"--> Fetch all the existing Data of DB To compair.*  
  SELECT \* FROM ZTSD\_PRODUCTLINE INTO TABLE @DATA(TEMP\_PRODUCTLINE).  
  SORT TEMP\_PRODUCTLINE BY BUSINESS\_ID VKORG PRDT\_LINEID.  
  CLEAR:LT\_PRODUCTLINE,LS\_PRODUCTLINE.  
  
  IF LT\_DATA is NOT INITIAL.  
*\*--> 4.1:Prepare the data to load in database table and send a success response*  
*\*        once data uploaded in database table.*  
  IF LS\_ENTITY-OPERATION = 'CR'.                             *"==>CR :- create Record.*  
   LOOP AT LT\_DATA INTO DATA(LS\_DATA).  
   READ TABLE TEMP\_PRODUCTLINE INTO DATA(WA\_DATA)  
   WITH KEY BUSINESS\_ID = LS\_DATA-BUSINESS\_ID  
            VKORG       = LS\_DATA-VKORG  
            PRDT\_LINEID = |{ LS\_DATA-PRDT\_LINEID ALPHA = IN }| BINARY SEARCH.  
    IF WA\_DATA IS INITIAL.  
     LS\_PRODUCTLINE             = CORRESPONDING #( LS\_DATA ).  
     LS\_PRODUCTLINE-MANDT       = SY-MANDT.  
     LS\_PRODUCTLINE-PRDT\_LINEID = |{ LS\_PRODUCTLINE-PRDT\_LINEID ALPHA = IN }|.  
     APPEND LS\_PRODUCTLINE TO LT\_PRODUCTLINE.  
    ENDIF.  
   CLEAR: LS\_DATA,LS\_PRODUCTLINE,WA\_DATA.  
   ENDLOOP.  
  
   IF LT\_PRODUCTLINE IS NOT INITIAL.  
   INSERT ZTSD\_PRODUCTLINE FROM TABLE LT\_PRODUCTLINE.  
   IF SY-SUBRC EQ 0.  
    LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
     exporting  
       IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>SUCCESS  
       IV\_MSG\_TEXT               = 'Records Created Successfully.'  
       IV\_ADD\_TO\_RESPONSE\_HEADER = ABAP\_TRUE  
     ).  
     ER\_ENTITY-DATA\_STRING = 'Records Created Successfully.'.  
   ELSE.  
    *"-->Business Exception for Failed to crete in SAP.*  
    LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
     EXPORTING  
       IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>ERROR  
       IV\_MSG\_TEXT               = 'Failed to Create Record in SAP System.'  
     ).  
    RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
     EXPORTING  
     textid            = /iwbep/cx\_mgw\_busi\_exception=>BUSINESS\_ERROR  
     message\_container = lo\_message.  
   ENDIF.  
   ELSE.  
    *"--> Business exception for Record alredy present in sap.*  
    LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
     EXPORTING  
       IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>ERROR             *" Message Type - defined by GCS\_MESSAGE\_TYPE*  
       IV\_MSG\_TEXT               = 'Requested Record alredy exist in SAP.' *" Message Text*  
     ).  
    RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
     EXPORTING  
     textid            = /iwbep/cx\_mgw\_busi\_exception=>BUSINESS\_ERROR  
     message\_container = lo\_message.  
   ENDIF.  
*\*--> 4.2: Update the Database record based on Operation Field Value,*  
*\*         value = UR then Update the Database based on the UI Value*  
*\*         value = UD then Update the Delation Flag based on the UI Value*  
*\*         Value = DR then Update the Delation flag to X based on the Status Field value of Entity.*  
  ELSEIF LS\_ENTITY-OPERATION = 'DR' OR  
         LS\_ENTITY-OPERATION = 'UR' OR  
         LS\_ENTITY-OPERATION = 'UD'. *"==>UR: Update Record,DR:Delete Record*  
   CLEAR: LS\_DATA,WA\_DATA,LV\_MESSAGE.  
   LOOP AT LT\_DATA INTO LS\_DATA.  
    READ TABLE TEMP\_PRODUCTLINE INTO WA\_DATA  
    WITH KEY BUSINESS\_ID = LS\_DATA-BUSINESS\_ID VKORG  = LS\_DATA-VKORG  
             PRDT\_LINEID = |{ LS\_DATA-PRDT\_LINEID ALPHA = IN }| BINARY SEARCH.  
     IF WA\_DATA IS NOT INITIAL.  
       LS\_PRODUCTLINE               = CORRESPONDING #( WA\_DATA ).  
       LS\_PRODUCTLINE-PRDT\_LINENAME = LS\_DATA-PRDT\_LINENAME.  
       LS\_PRODUCTLINE-STATUS        = LS\_DATA-STATUS.  
       APPEND LS\_PRODUCTLINE TO LT\_PRODUCTLINE.  
     ENDIF.  
     CLEAR: LS\_DATA,LS\_PRODUCTLINE,WA\_DATA.  
   ENDLOOP.  
  
   IF LT\_PRODUCTLINE IS NOT INITIAL.  
    MODIFY ZTSD\_PRODUCTLINE FROM TABLE LT\_PRODUCTLINE.  
    IF SY-SUBRC EQ 0.  
     IF LS\_ENTITY-OPERATION = 'UR'.  
       LV\_MESSAGE = 'Record Updated Successfully.'.  
     ELSEIF LS\_ENTITY-OPERATION = 'DR'.  
       LV\_MESSAGE = 'Record Deleted Successfully.'.  
     ELSEIF LS\_ENTITY-OPERATION = 'UD'.  
       LV\_MESSAGE = 'Delation Status Removed Successfully.'.  
     ENDIF.  
      LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
      exporting  
       IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>SUCCESS   *" Message Type - defined by GCS\_MESSAGE\_TYPE*  
       IV\_MSG\_TEXT               = LV\_MESSAGE*" Message Text*  
       IV\_ADD\_TO\_RESPONSE\_HEADER = ABAP\_TRUE                       *" Flag for adding or not the message to the response header*  
     ).  
     ER\_ENTITY-DATA\_STRING = LV\_MESSAGE.  
    ELSE.  
      *"-->Business Exception for Failed to crete in SAP.*  
      IF LS\_ENTITY-OPERATION = 'UR'.  
       LV\_MESSAGE = 'Failed to Update Record in SAP System.'.  
      ELSEIF LS\_ENTITY-OPERATION = 'DR'.  
       LV\_MESSAGE = 'Failed to Delete Record in SAP System.'.  
     ELSEIF LS\_ENTITY-OPERATION = 'UD'.  
       LV\_MESSAGE = 'Delation Status Removed Successfully.'.  
      ENDIF.  
      LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
      EXPORTING  
       IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>ERROR              *" Message Type - defined by GCS\_MESSAGE\_TYPE*  
       IV\_MSG\_TEXT               = 'Failed to Update Record in SAP System.' *" Message Text*  
       ).  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
      EXPORTING  
       textid            = /iwbep/cx\_mgw\_busi\_exception=>BUSINESS\_ERROR  
       message\_container = lo\_message.  
    ENDIF.  
   ELSE.  
    *"--> Business exception for Record alredy present in sap if in the request no new record fornd.*  
    LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
     EXPORTING  
       IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>ERROR  
       IV\_MSG\_TEXT               = 'Requested Record not exist in SAP system.'  
     ).  
    RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
     EXPORTING  
     textid            = /iwbep/cx\_mgw\_busi\_exception=>BUSINESS\_ERROR  
     message\_container = lo\_message.  
   ENDIF.  
  ENDIF.  
  ENDIF.  
  
  ELSE.  
    *"--> Raise Exception if Operation Code is Worng.*  
    LO\_MESSAGE->ADD\_MESSAGE\_TEXT\_ONLY(  
     EXPORTING  
       IV\_MSG\_TYPE               = /iwbep/cl\_cos\_logger=>ERROR  
       IV\_MSG\_TEXT               = 'Enter Correct Operation Code(CR/UP/DR/UD) in Payload.'  
     ).  
    RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
     EXPORTING  
     textid            = /iwbep/cx\_mgw\_busi\_exception=>BUSINESS\_ERROR  
     message\_container = lo\_message.  
  ENDIF.  
  ENDMETHOD.

**Step 4: Register the service using “/N/IWFND/MAINT\_SERVICE”**

Service name: ZGW\_SD\_PRODUCTLINE\_SRV



**4.Testing API with Payload and Testing screen’s Screenshot.**

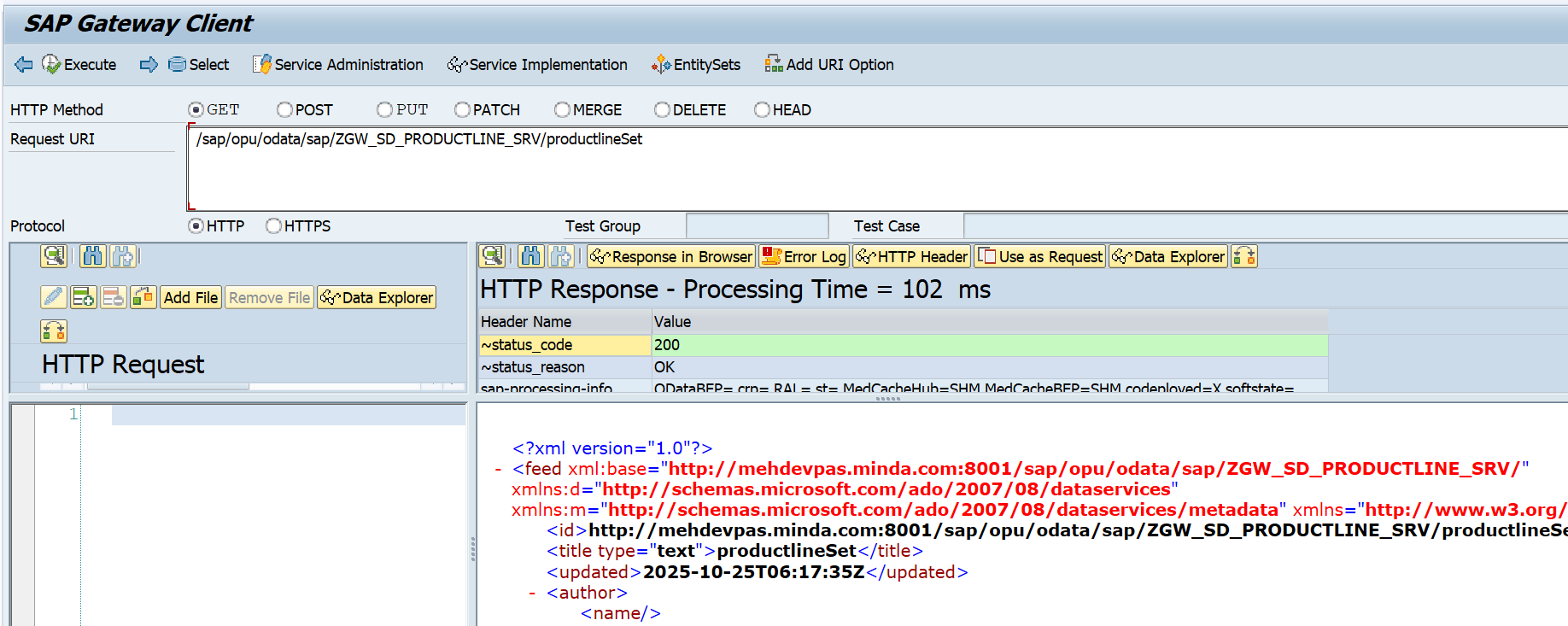
Tcode used for Testing: **/n/iwfnd/gw\_client**

**Case 1: Get the Entity or Entity Set Details.**

**1.1: Get All the record from Database table.**

**SAP Gateway Client**

API: /sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet

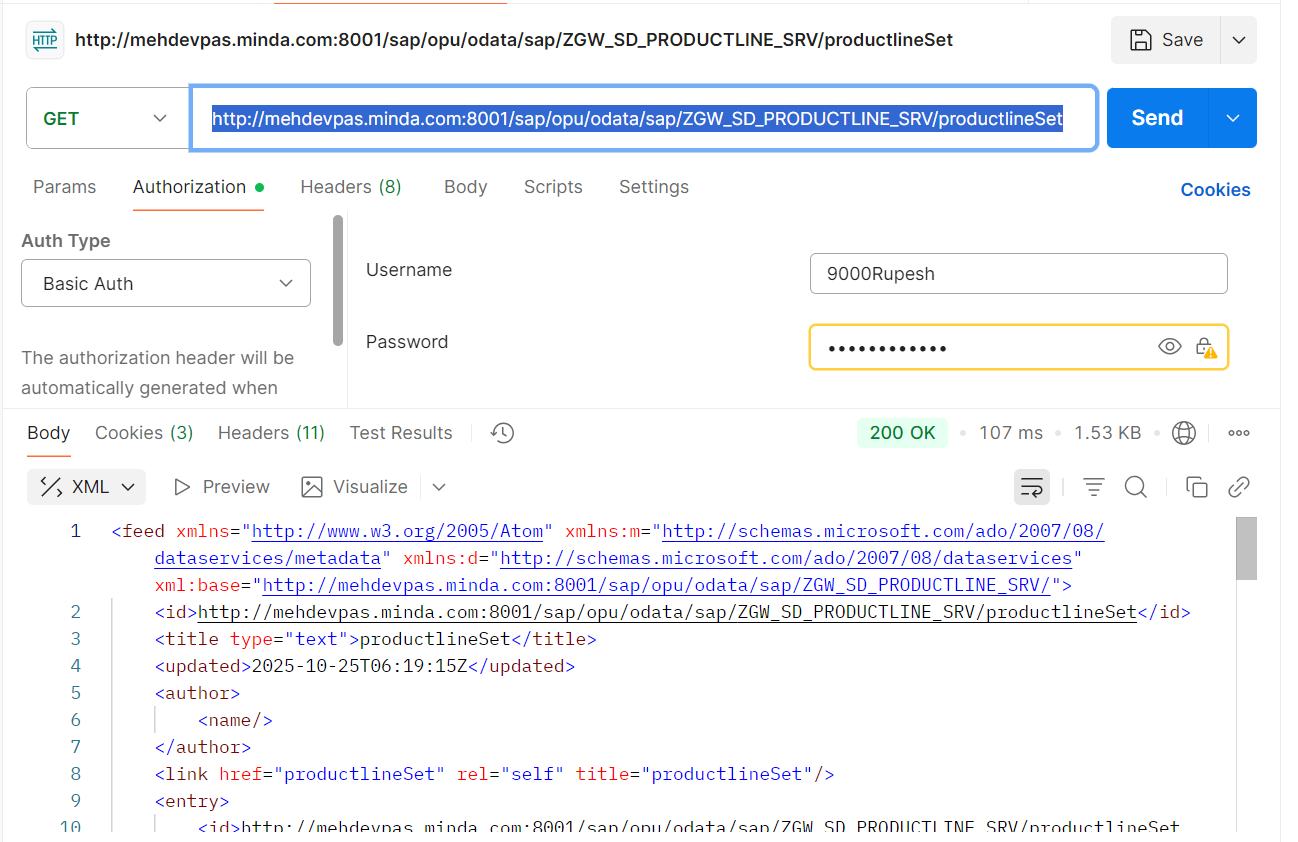


**Postman Testing**

When we are dealing with post man or outside SAP application through SAP API, we have to pass and Valid sap user is and password in the Authorization section.

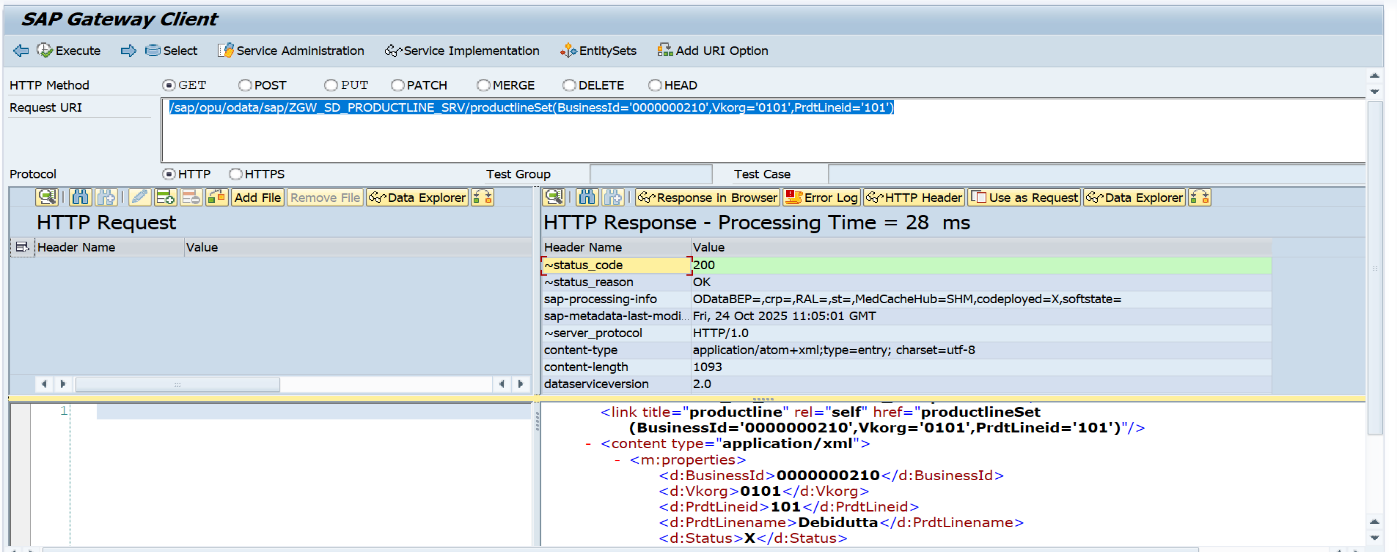
If you are new to Postman the follow the step for, GET:

Goto postman🡪 Pest /write URL🡪 SET the Methos(GET)🡪Authorization(In “Auth type” pass “Basic Auth”🡪 In the User id & password section pass valid user iD and password )🡪 click on Send.

API: http://mehdevpas.minda.com:8001/sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet

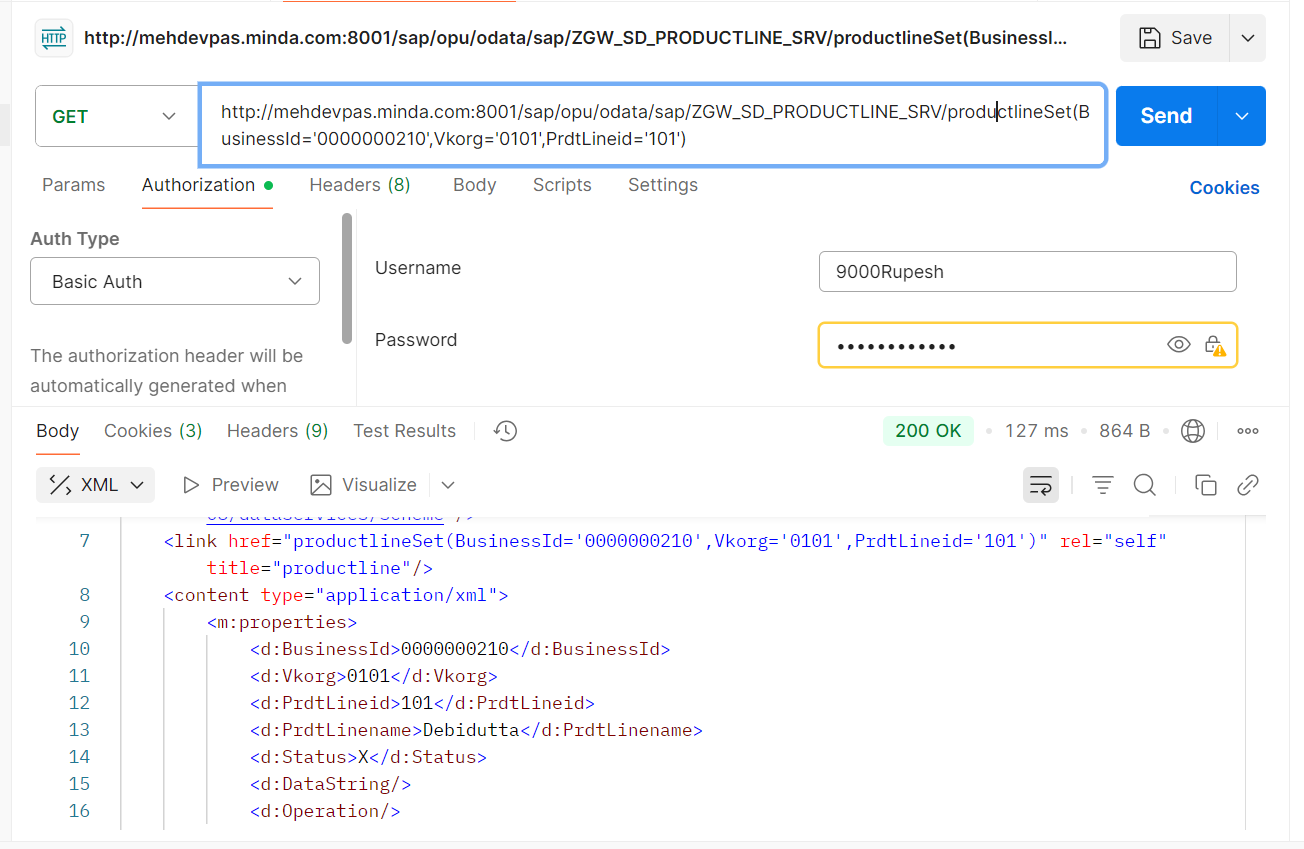
**1.2: Fetch Single record from Database**

**SAP Gateway client:**

API: /sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet(BusinessId='0000000210',Vkorg='0101',PrdtLineid='101')

**Postman Testing**

API: <http://mehdevpas.minda.com:8001/sap/opu/odata/sap/ZGW_SD_PRODUCTLINE_SRV/productlineSet(BusinessId='0000000210',Vkorg='0101',PrdtLineid='101’)>



**Case 2: Create/Update/Delete Operation :**

Before entering into CUD operation, please gone through the Note which will help you to understand the while doing operation through Postman and Gateway.

while we doing These operation we have to provide CSRF Token in the “Header Request” section for verified communication as well as security measure. Using a Cross-Site Request Forgery (CSRF) token in SAP OData is a security measure to prevent unauthorized and malicious commands from being executed on behalf of an authenticated user. It is required for any request that modifies data on the server, such as POST (create), PUT/PATCH (update), and DELETE.

**How to get CSRF token?**

While dealing through SAP Gateway client, no need to fetch and Provide any CSRF Token While performing CRUD because by default system handle all these things.

But if you are trying to handle Create, update and delete from outside SAP system, then we need to handle the CSRF Token Manually in the Header Request along with SAP User ID & Password. In the below postman testing you can check .

Here I have divided the CRUD testing in two parts

1. Gateway Testing (No need to handle the CSRF token Part manually)
2. Postman Testing (Here we have to handle the CSRF token Manually)

**Note:**

Before start the operation, we have to follow the below details strictly.

1. DataString – This field carries the mass data in a stringified JSON format.
2. Operation – This field specifies the type of operation to be performed (CR for Create, UR for Update, DR for Set Status as X/Delete, and UD for Remove the Status from X/undo Delete).Don’t use other Operation Code, other wise SAP System raise Business Exception With Valid Error message.
3. HTTP Method Always ‘POST’.

**A: Gateway Testing (No need to handle the CSRF token Part manually)**

**2.1 Create Operation**

API: /sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet

Payload used:

{

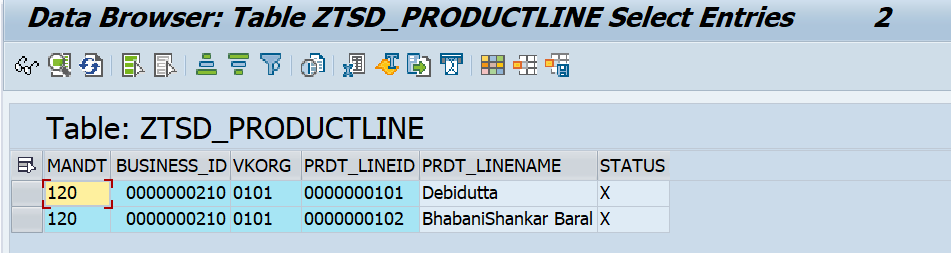
"DataString": "[{\"BusinessId\":\"0000000310\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000101\",\"PrdtLinename\":\"UNO Minda\",\"Status\":\"\"},{\"BusinessId\":\"0000000311\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000102\",\"PrdtLinename\":\"AIRDIT Software\",\"Status\":\"\"},{\"BusinessId\":\"0000000312\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000103\",\"PrdtLinename\":\"SAP Germany\",\"Status\":\"\"}]",

"Operation": "CR"

}

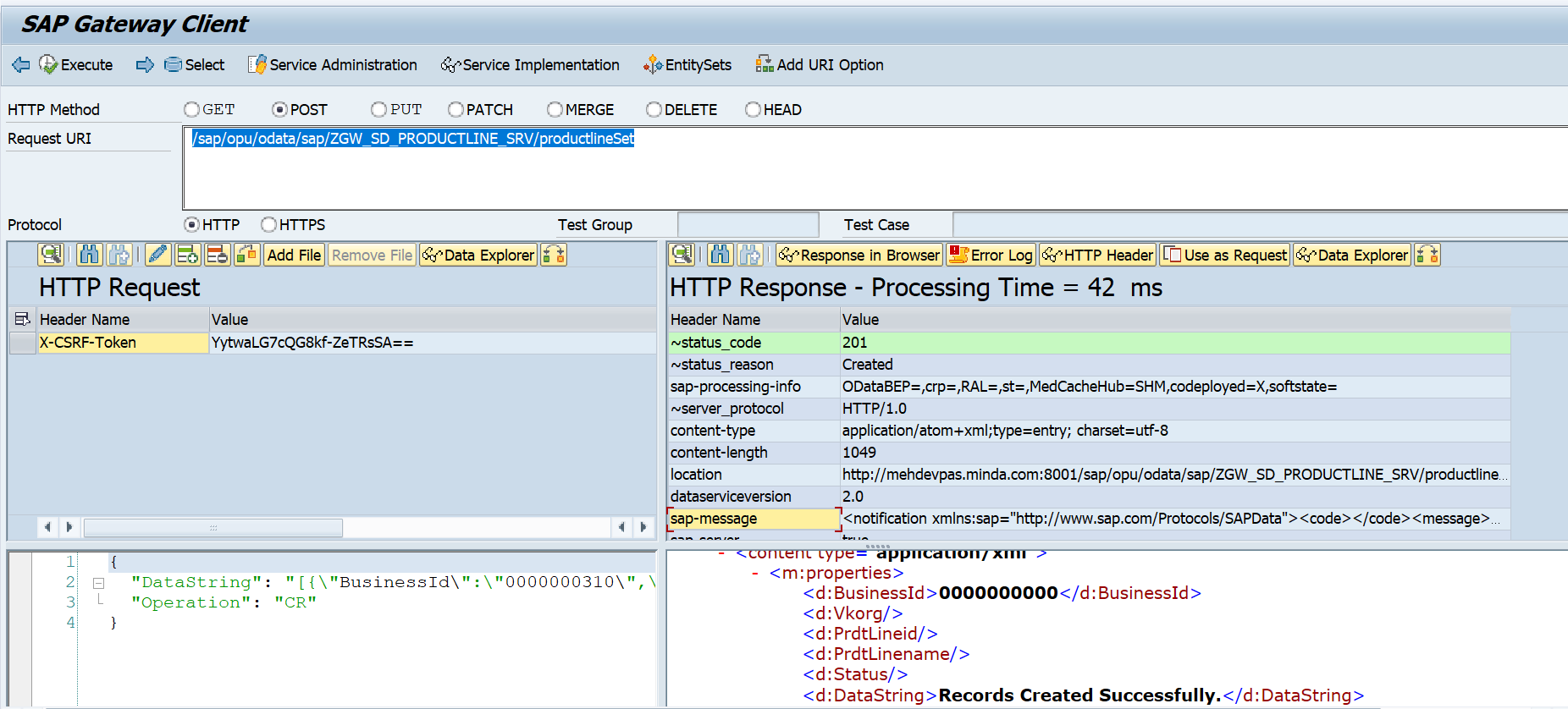
**Positive testing**

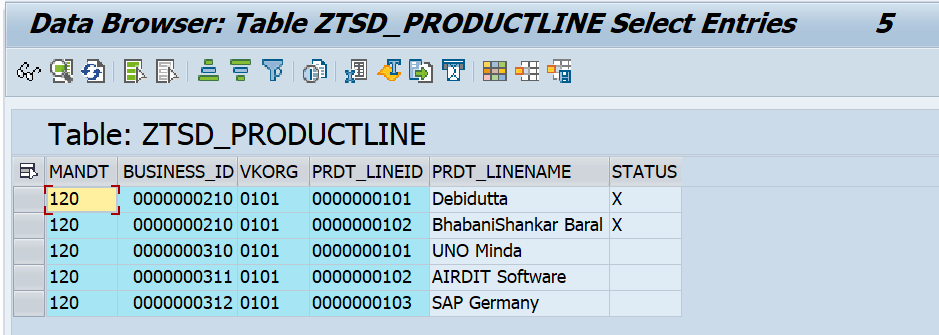
If Header Response’s status code is ‘201’, the record successfully Crated in Database table.

**Before click execute, ZTSD\_PRODUCTLINE Database Table.**

**Click on Execute,Gateway screen with success status.**

Here you will get the success message in two pace

* Header response body’s DataString Field
* ****Header response’s sap-message Field.

**After click execute, ZTSD\_PRODUCTLINE Database Table.**

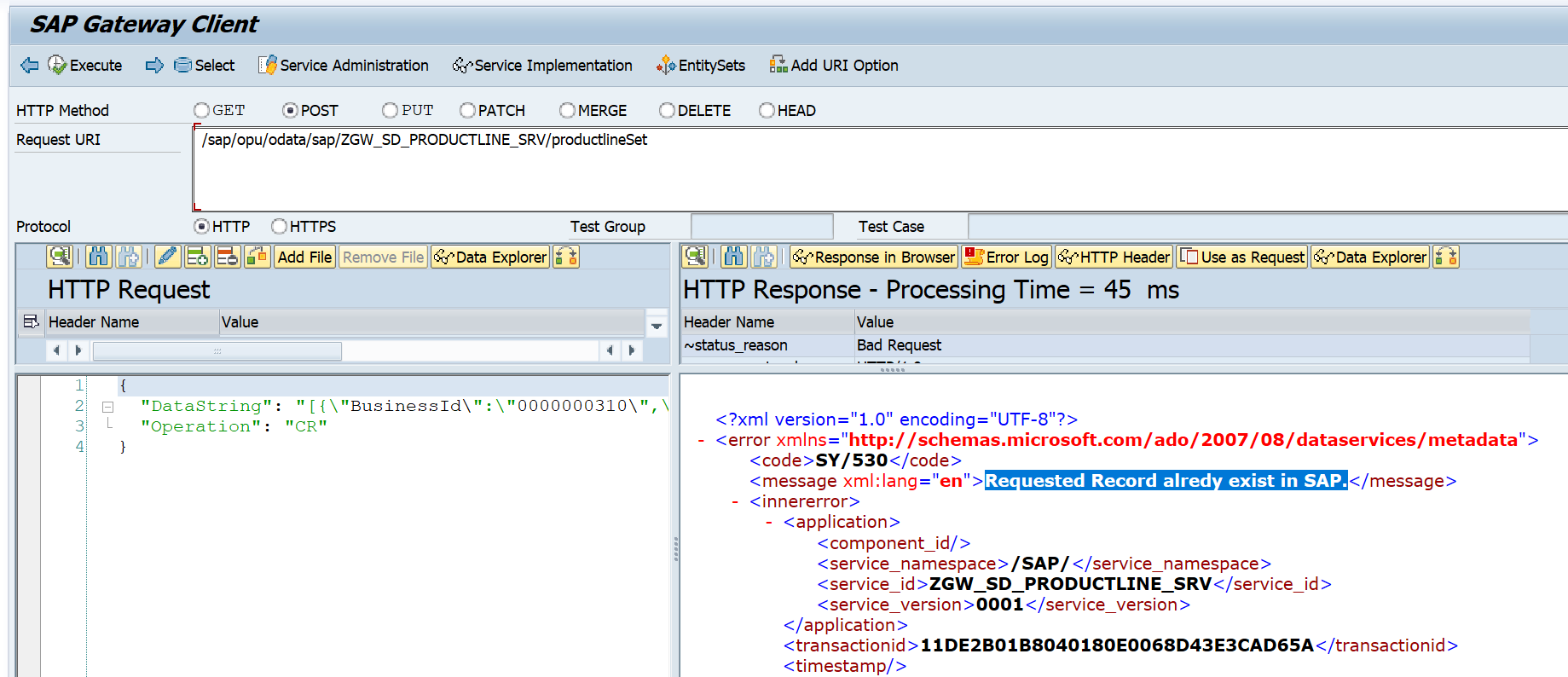
**Note: if user entering some existing data(already present in SAP) along with new data then it will ignore the existing data & insert the new record in database table .**

**Negative Testing:**

**Case1:**

If User Trying to enter the existing data again in the payload, then following error message will come with business exception in Response Body.

Message: Requested Record already exist in SAP.



**Case 2:**

If, for any reason, record is not created in the SAP system, the system will raise a business exception as an error. The error message displayed will be the same as in the negative case mentioned above.

Message: Failed to Create Record in SAP System.

**2.2: Update Operation**

API: /sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet

Payload used:

{

"DataString": "[{\"BusinessId\":\"0000000310\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000101\",\"PrdtLinename\":\"UNO Minda Group IT\",\"Status\":\"\"},{\"BusinessId\":\"0000000311\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000102\",\"PrdtLinename\":\"AIRDIT Software PVT LTD\",\"Status\":\"\"},{\"BusinessId\":\"0000000312\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000103\",\"PrdtLinename\":\"SAP Labs India\",\"Status\":\"\"}]",

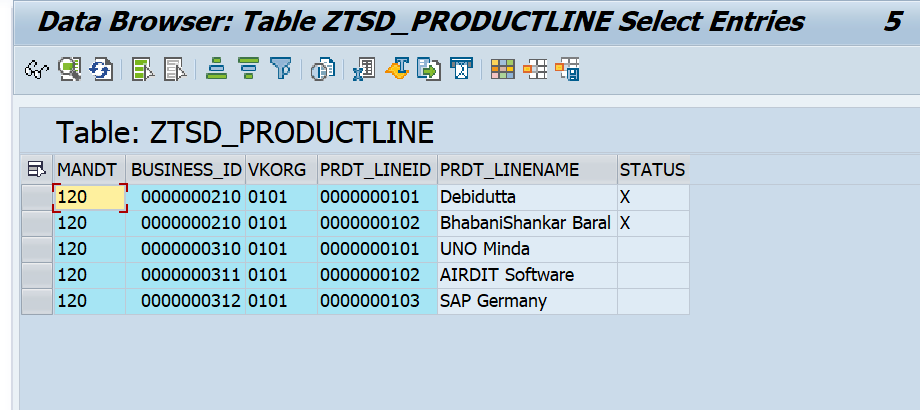
"Operation": "UR"

}

**Positive testing**

If Header Response’s status code is ‘201’, the record successfully Updated in Database table.

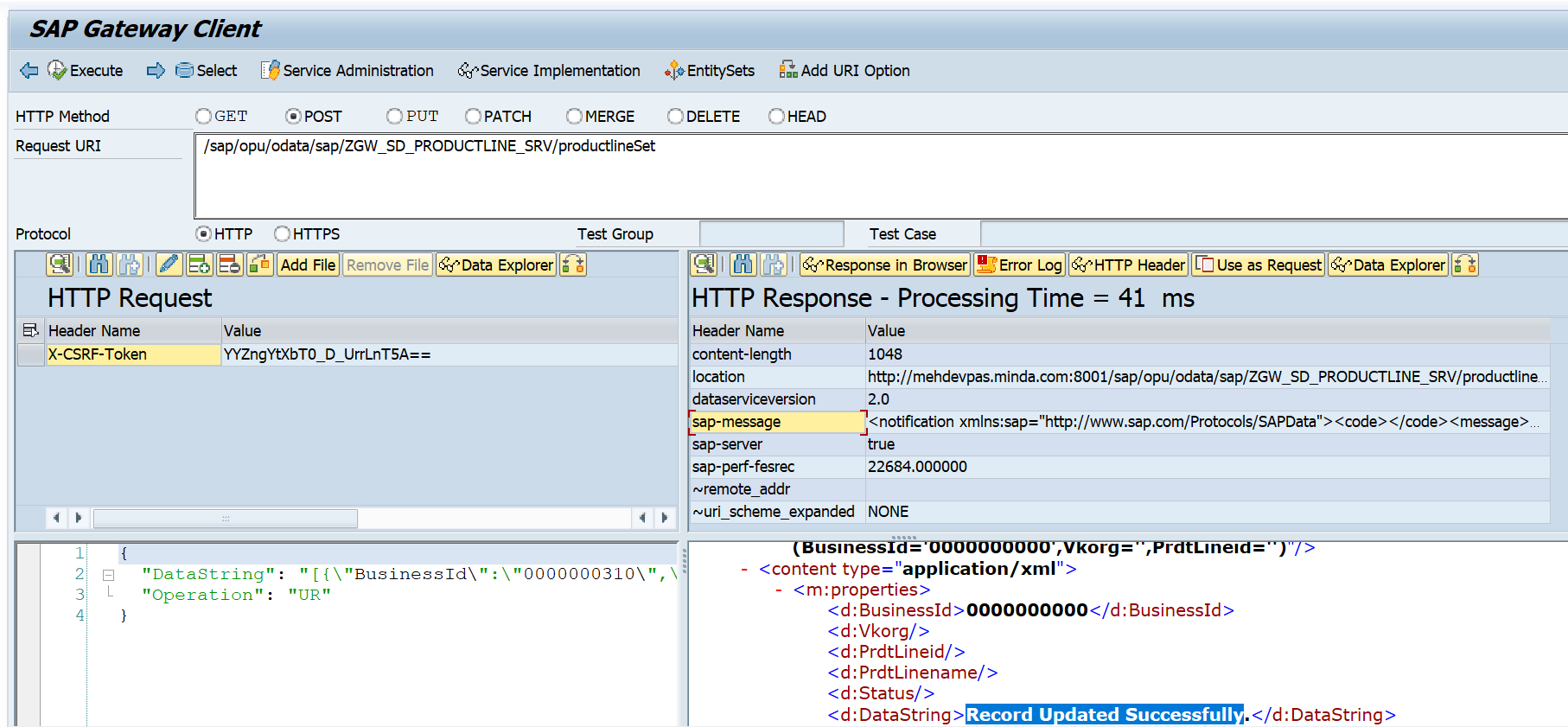
**Before click execute, ZTSD\_PRODUCTLINE Database Table.**



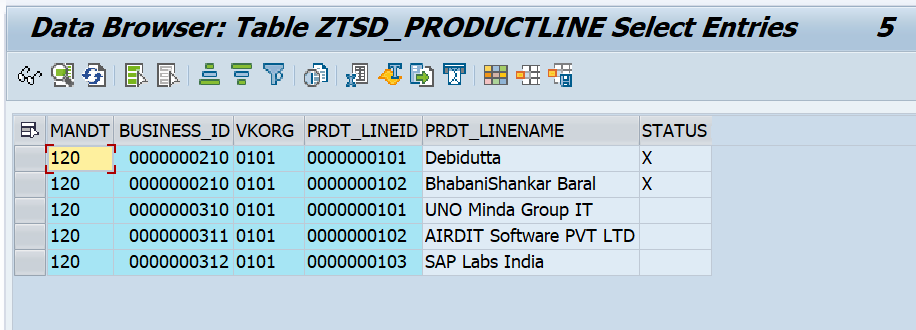
**Click on Execute,Gateway screen with success status.**

Here you will get the success message (Record Updated Successfully) in two pace

* Header response body’s DataString Field
* Header response’s sap-message Field.



**After click execute, ZTSD\_PRODUCTLINE Database Table.**

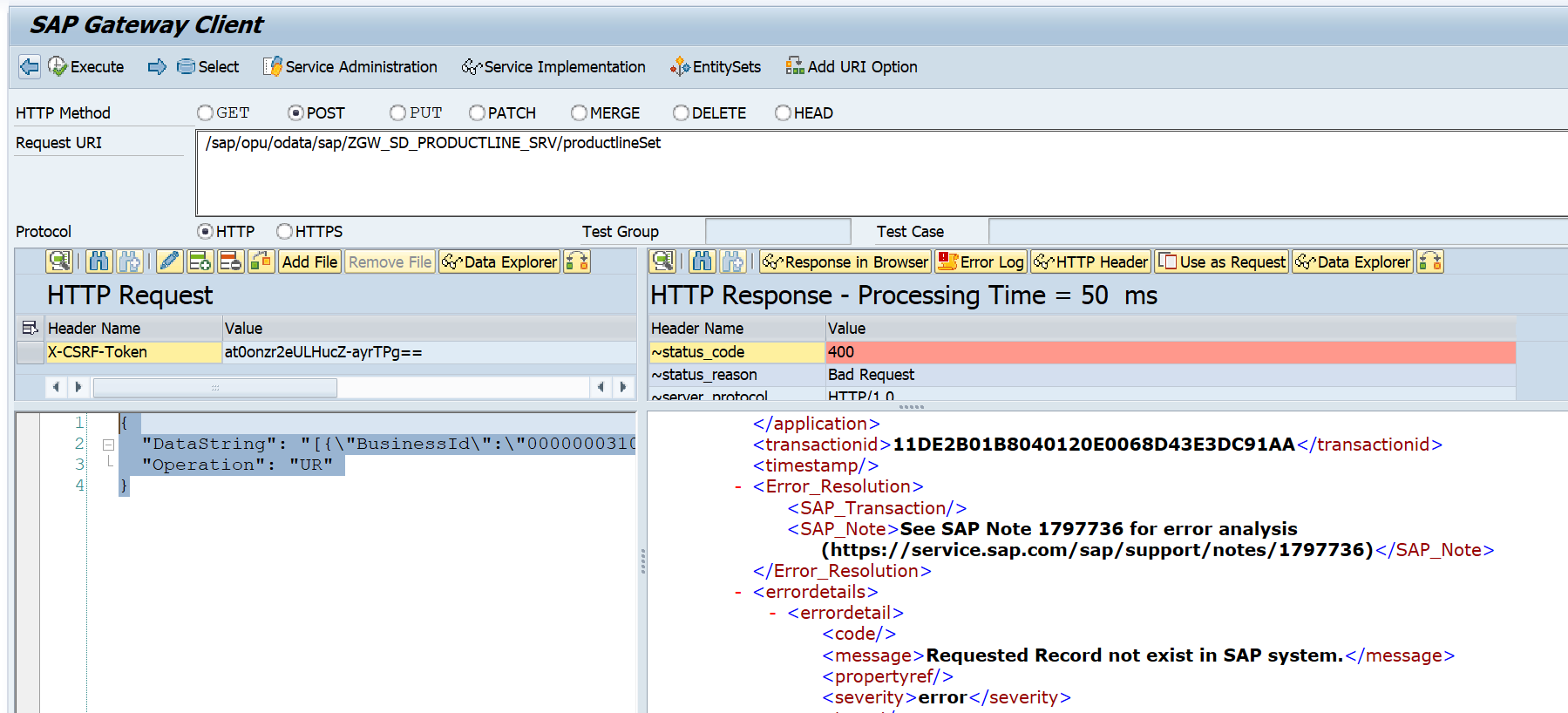


**Negative Testing:**

**Case1:**

If user want to update a record which is not present in SAP system.

Message: Requested Record not exist in SAP system.



**Case 2:**

If, for any reason, record is not Updated in the SAP system, the system will raise a business exception as an error. The error message displayed will be the same as in the negative case mentioned above.

Message: Failed to Update Record in SAP System.

**2.3: Delete Operation**

API: /sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet

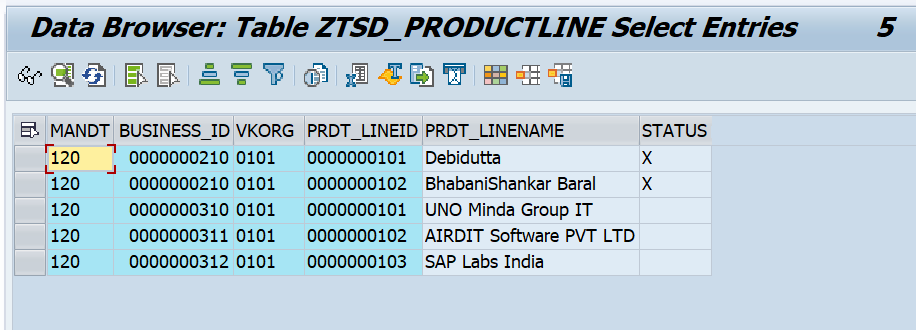
Payload used:

{  
  "DataString": "[{\"BusinessId\":\"0000000310\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000101\",\"PrdtLinename\":\"UNO Minda Group IT\",\"Status\":\"X\"},{\"BusinessId\":\"0000000311\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000102\",\"PrdtLinename\":\"AIRDIT Software PVT LTD\",\"Status\":\"X\"},{\"BusinessId\":\"0000000312\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000103\",\"PrdtLinename\":\"SAP Labs India\",\"Status\":\"X\"}]",  
  "Operation": "DR"  
}

**Positive testing**

If Header Response’s status code is ‘201’, the record successfully Updated in Database table.

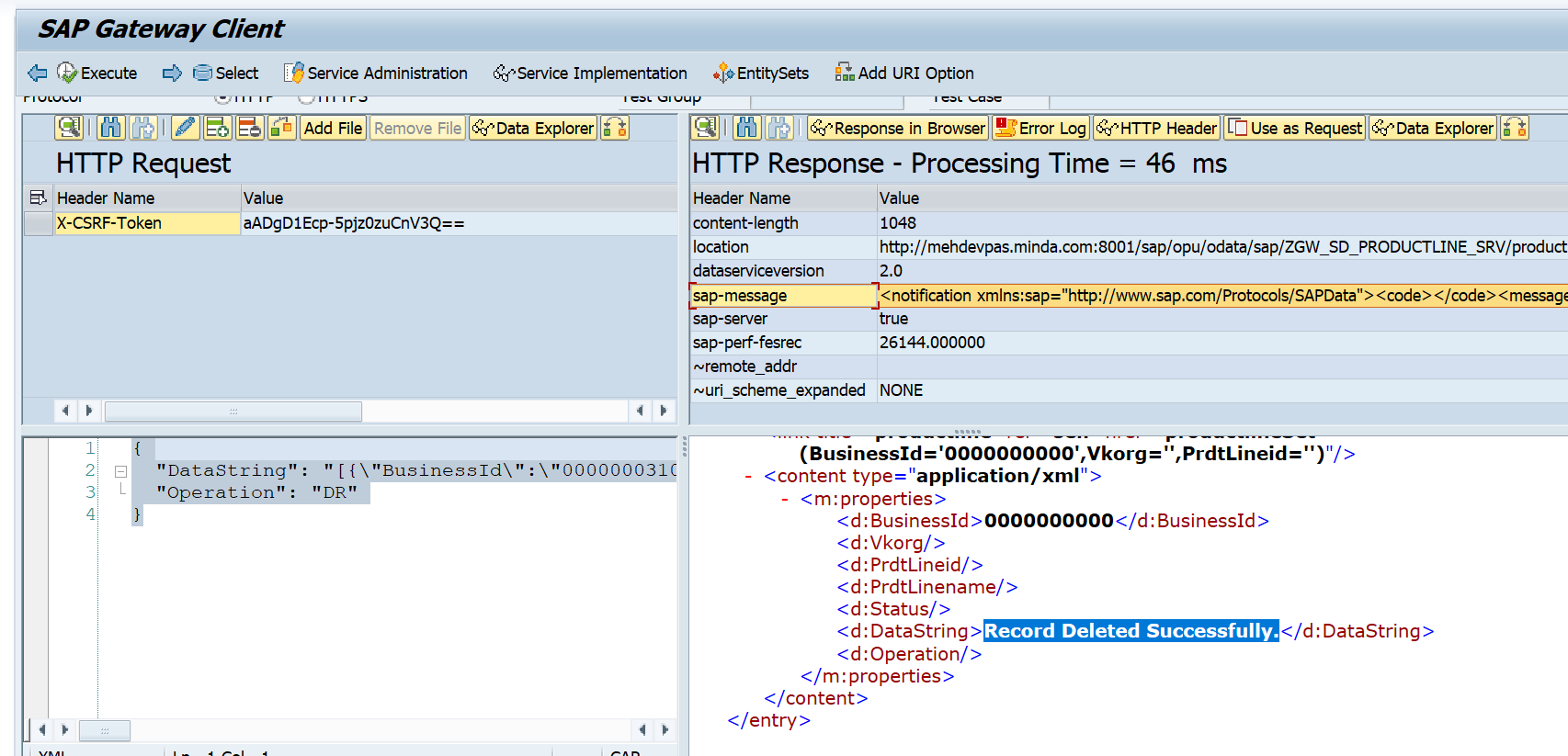
**Before click execute, ZTSD\_PRODUCTLINE Database Table.**



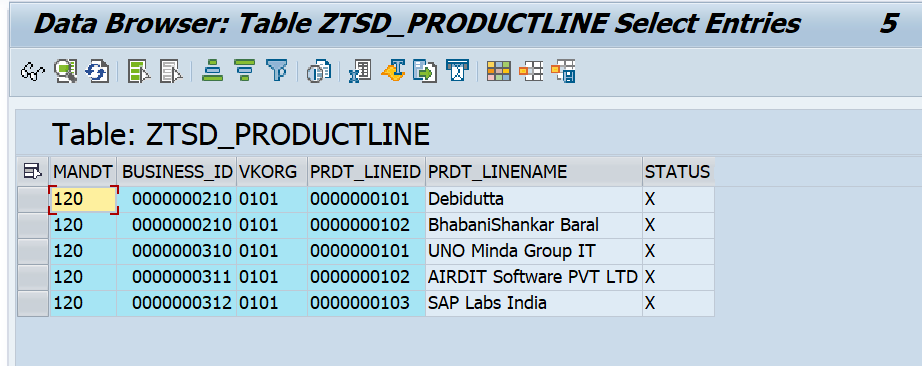
**Click on Execute,Gateway screen with success status.**

Here you will get the success message (Record Deleted Successfully.) in two pace

* Header response body’s DataString Field
* Header response’s sap-message Field.



**After click execute, ZTSD\_PRODUCTLINE Database Table.**

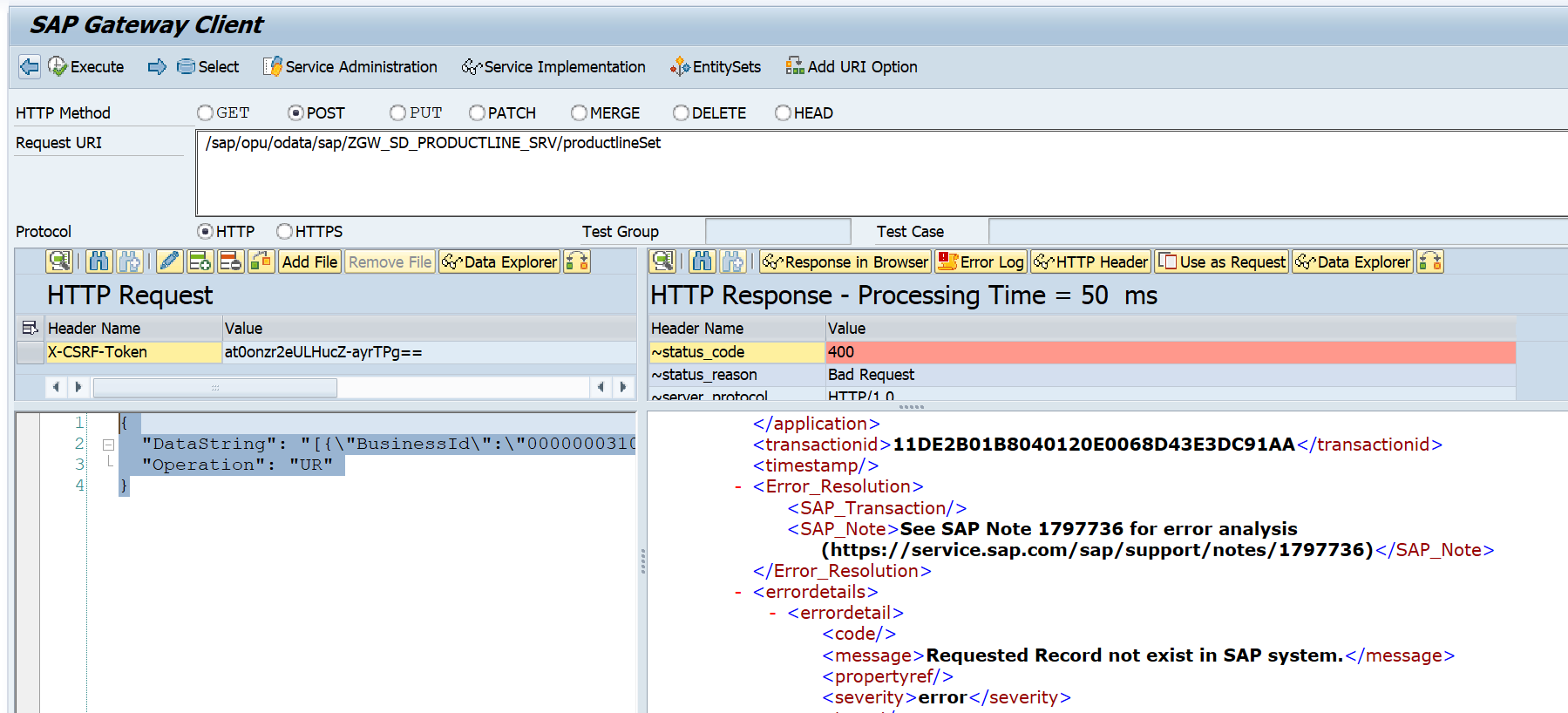


**Negative Testing:**

**Case1:**

If user want to update a record which is not present in SAP system.

Message: Requested Record not exist in SAP system.



**Case 2:**

If, for any reason, record is not Updated in the SAP system, the system will raise a business exception as an error. The error message displayed will be the same as in the negative case mentioned above.

Message: Failed to Delete Record in SAP System.

**2.2: Update Operation**

API: /sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet

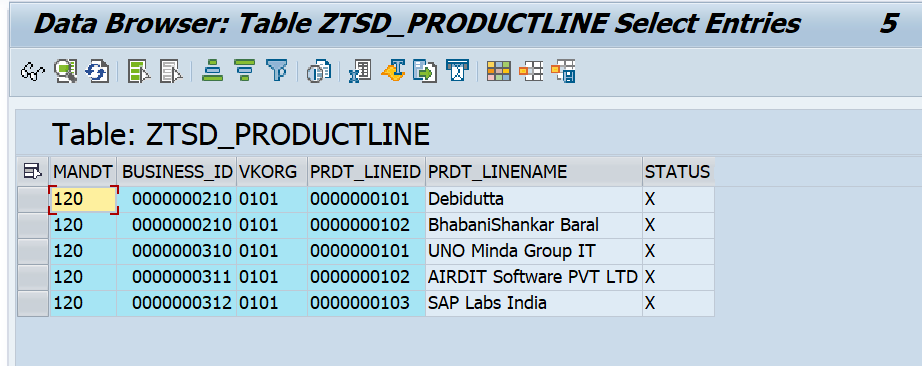
Payload used:

{  
  "DataString": "[{\"BusinessId\":\"0000000310\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000101\",\"PrdtLinename\":\"UNO Minda Group IT\",\"Status\":\"\"},{\"BusinessId\":\"0000000311\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000102\",\"PrdtLinename\":\"AIRDIT Software PVT LTD\",\"Status\":\"\"},{\"BusinessId\":\"0000000312\",\"Vkorg\":\"0101\",\"PrdtLineid\":\"0000000103\",\"PrdtLinename\":\"SAP Labs India\",\"Status\":\"\"}]",  
  "Operation": "UD"  
}

**Positive testing**

If Header Response’s status code is ‘201’, the record successfully Updated in Database table.

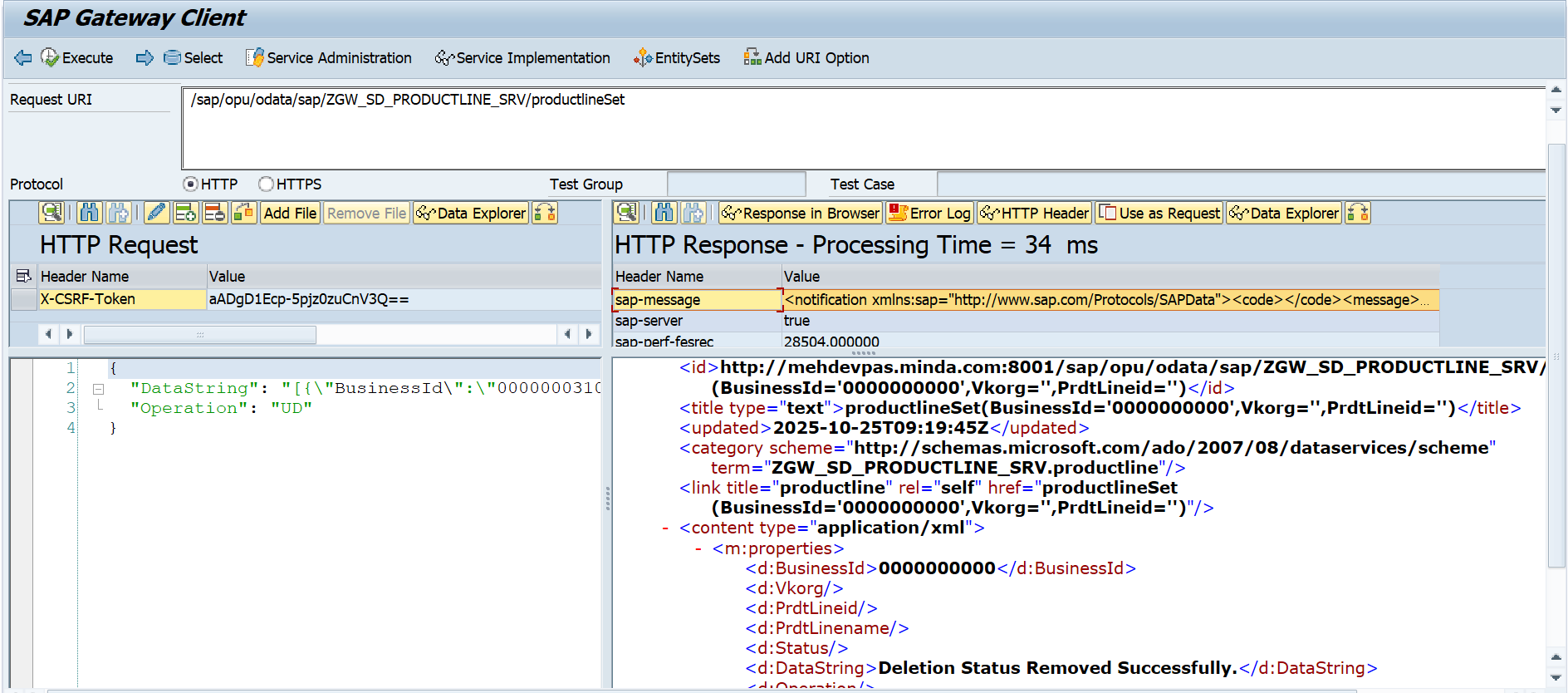
**Before click execute, ZTSD\_PRODUCTLINE Database Table.**



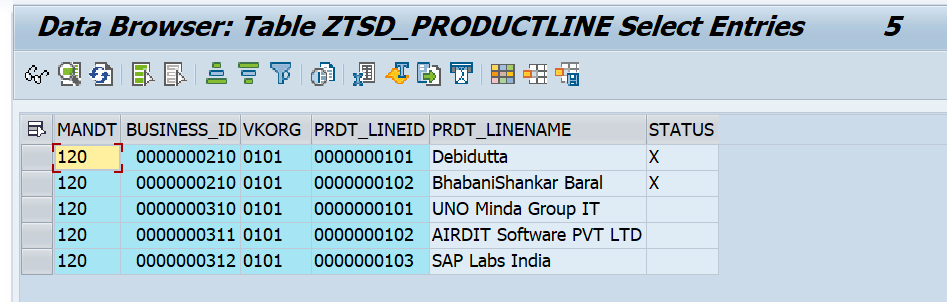
**Click on Execute,Gateway screen with success status.**

Here you will get the success message (Deletion Status Removed Successfully) in two pace

* Header response body’s DataString Field
* Header response’s sap-message Field.



**After click execute, ZTSD\_PRODUCTLINE Database Table.**

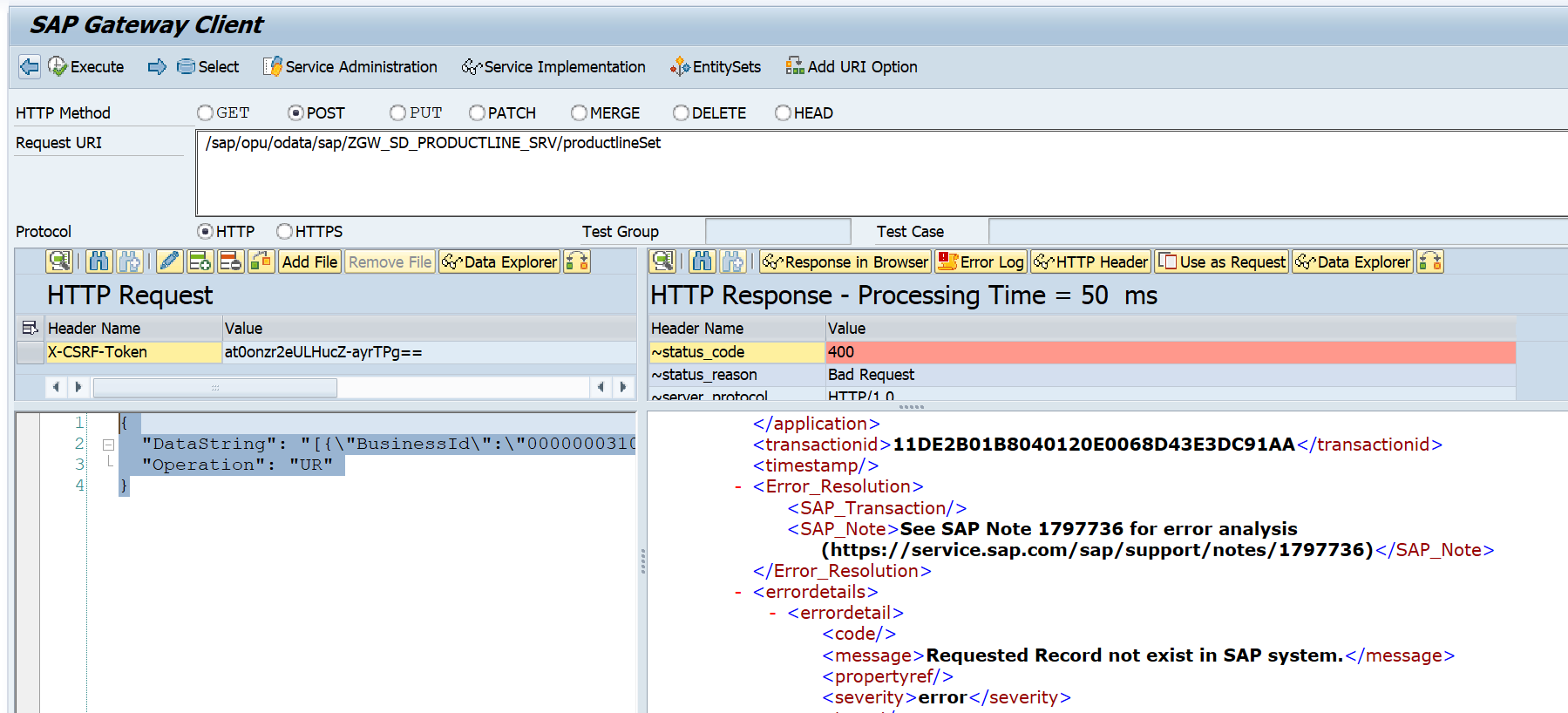


**Negative Testing:**

**Case1:**

If user want to update a record which is not present in SAP system.

Message: Requested Record not exist in SAP system.



**Case 2:**

If, for any reason, record is not Updated in the SAP system, the system will raise a business exception as an error. The error message displayed will be the same as in the negative case mentioned above.

Message: Failed to Remove Deletion Flag.

**C: Postman Testing (Here we have to handle the CSRF token Manually)**

API: http://mehdevpas.minda.com:8001/sap/opu/odata/sap/ZGW\_SD\_PRODUCTLINE\_SRV/productlineSet

As we know postman is a Outside SAP System, so here we have to handle the CSRF Token Manually.

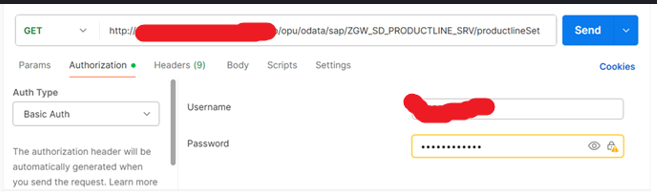
**How to Get CSRF Token?**

Follow the below steps:

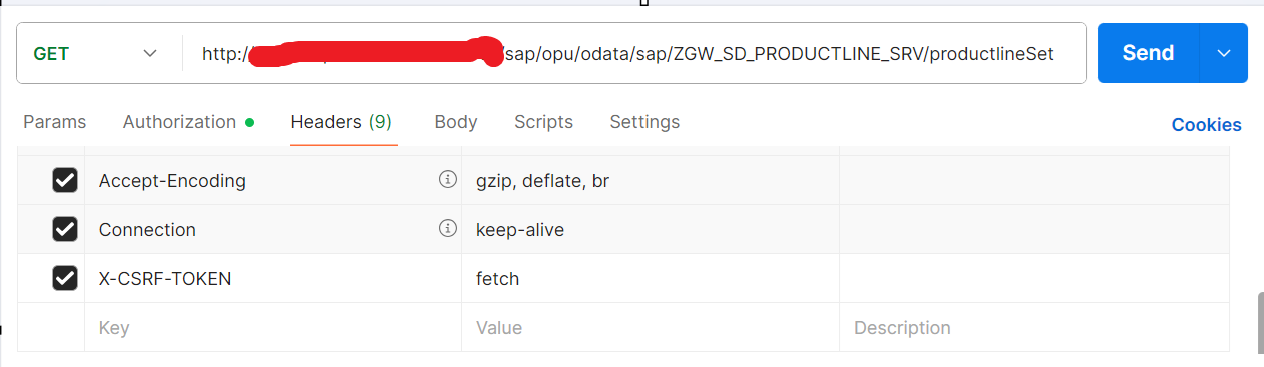
1.Go to postman

2.Pass the URL & pass Request Method as ‘GET’.

3.pass the Authorization detail (Authorization->Auth Type->Basic auth->SAP User ID & Password).

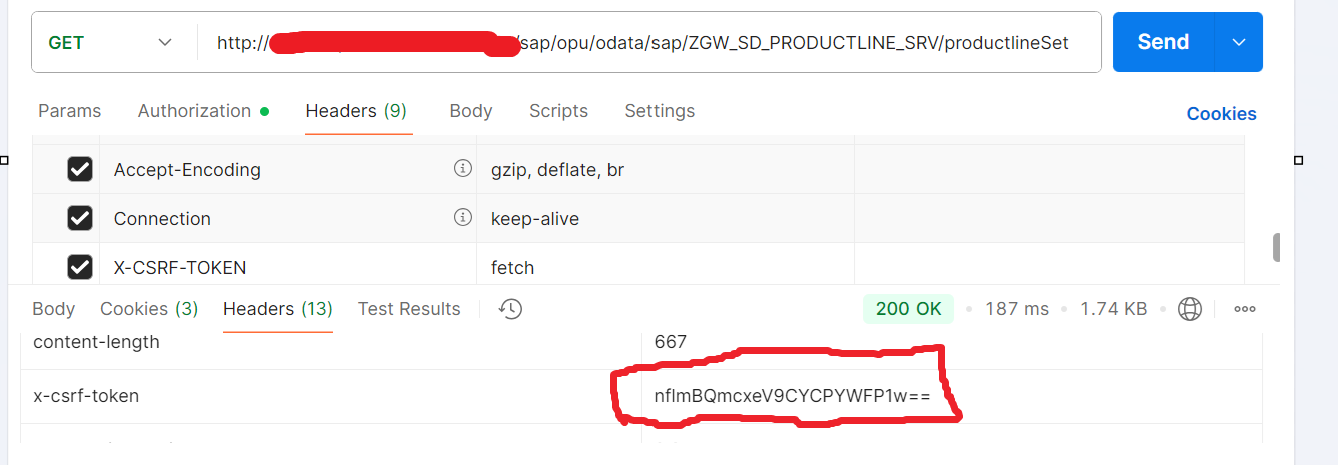


4.Then in the request Header section Pass the KEY = ‘X-CSRF-TOKEN’ & Value = ‘fetch’



5.Click on Send.

6.IF the Response is 200 OK (Request successful. The server has responded as required.), then go to the Response Header and check the “x-csrf-token” Value and use that value during all the Create/Update/Delete Operation.

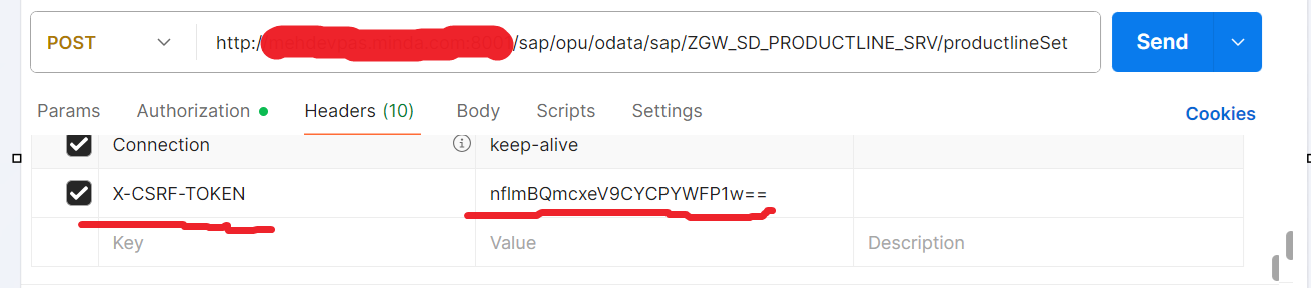


**Handle Crete Operation Through Postman:**

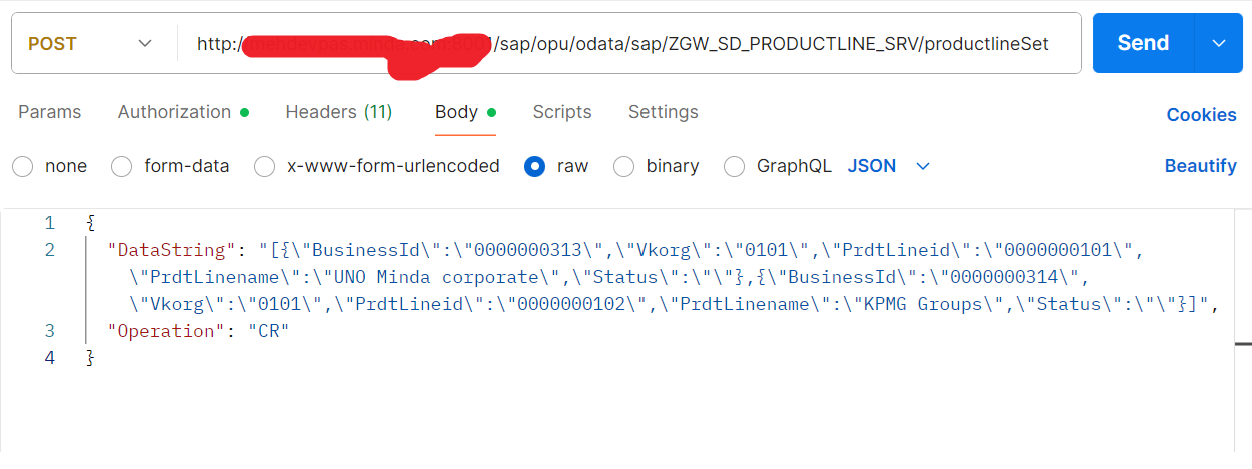
Step 1: Pass the URL & set Request method as “POST”.

Step 2: Handel the Authorization detail (Authorization->Auth Type->Basic auth->SAP User ID & Password).

Step 3: Pass the CSRF Token, which was fetch by the Above process like below.

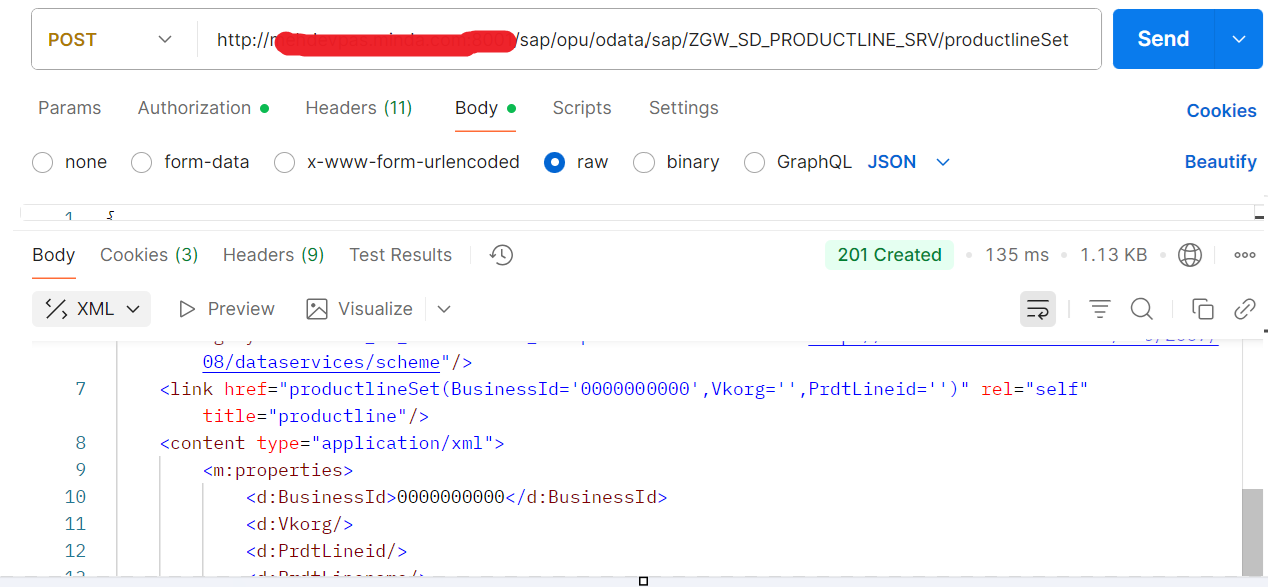


Step 4: Pass the Payload in Json Formart(Header Request🡪body🡪raw section)

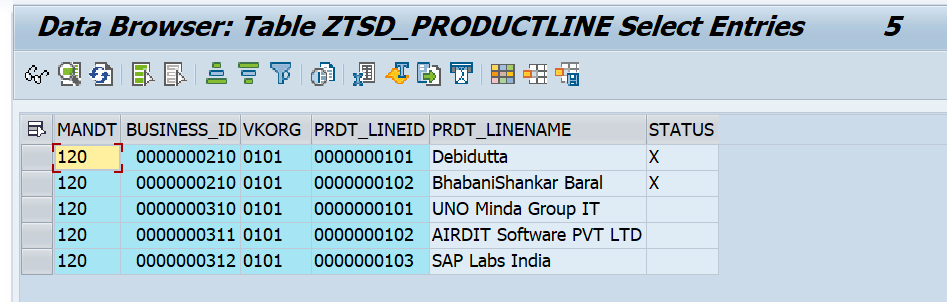


Step 5: click on Send.

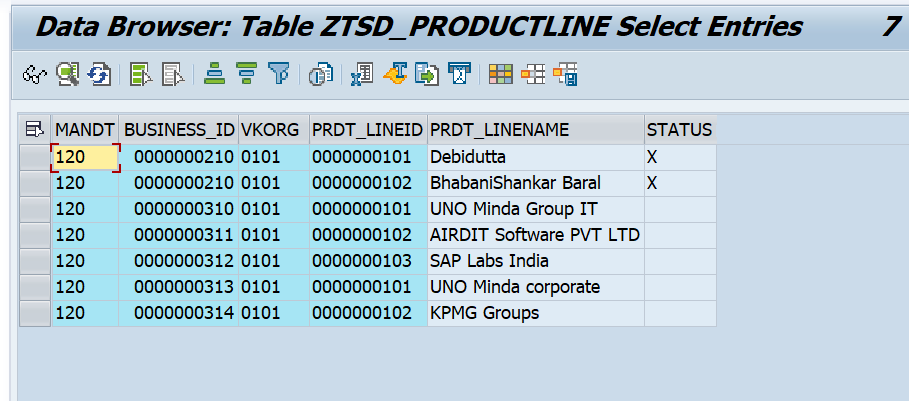
Step 6: if response status is 201, then record successfully created. Also check the Response body’s “DataString” field s well as Response header’s sap-message Value.



**Before click send.**



**After click send.**



Note: In the Same way you can check the Update and Delete part through Outside SAP system.