

Software Architecture

3rd Year – Semester 01

Enterprise Application Integration

Lecturer - By Udara Samaratunge

Task 01 (Access Application Interfaces)

Search **Google Oauth 2.0 playground** and select Application as below. Then select the API and click **Authorize APIs** as below.

OAuth 2.0 Playground ✕

▼ Step 1 Select & authorize APIs

Select the scope for the APIs you would like to access or input your own OAuth scopes below. Then click the "Authorize APIs" button.

Cloud SQL Administration API v1beta3

▶ Cloud Storage API v1

▶ Compute Engine API v1

▼ Contacts v3

- ✓ <https://www.google.com/m8/feeds/>

▶ Content API for Shopping v2

▶ DCM/DFA Reporting And Trafficking API v2.0

▶ Deployment Manager API v1beta2

▶ DoubleClick Search API v2

Authorize APIs

Request / Response

POST /o/oauth2/token HTTP/1.1
Host: accounts.google.com
Content-length: 163
content-type: application/x-www-form-urlencoded
user-agent: google-oauth-play

client_secret=*****&grant_type=authorization_code&client_id=407408718192.apps.googleusercontent.com

HTTP/1.1 200 OK
Alternate-protocol: 443:quic,
Content-length: 158
X-xss-protection: 1; mode=block
X-content-type-options: nosniff
Content-disposition: attachment
Expires: Fri, 01 Jan 1990 00:00:00 GMT
X-google-cache-control: remote-cache
Server: GSE
Via: HTTP/1.1 GWA
Pragma: no-cache
Cache-control: no-cache, no-store, max-age=0, must-revalidate
Date: Sun, 25 Jan 2015 15:32:32 GMT
X-frame-options: SAMEORIGIN
Content-type: application/json

- Then in Step 2 section click “**Exchange authorization token**” button as below.
- You can refresh click **Refresh token** when it is going to expire.
- It generate **Access token**. Copy that to access the API.
- Then copy that access token and use that for Postman Package and send API request as below for Google contact API.

OAuth 2.0 Playground
✕

▶ Step 1 Select & authorize APIs

▼ Step 2 Exchange authorization code for tokens

Once you got the Authorization Code from Step 1 click the **Exchange authorization code for tokens** button, you will get a refresh and an access token which is required to access OAuth protected resources.

Authorization code:

Exchange authorization code for tokens

Refresh token:

Access token: Refresh access token

☐ Auto-refresh the token before it expires.

The access token will expire in 3590 seconds.

Request / Response

```
POST /o/oauth2/token HTTP/
Host: accounts.google.com
Content-length: 265
content-type: application/
user-agent: google-oauth-p

code=4%2FeXw3_G5yzocRBTZzq
elopers.google.com%2Foauth
****&grant_type=authoriza

HTTP/1.1 200 OK
Alternate-protocol: 443:qu
Content-length: 248
X-xss-protection: 1; mode=
X-content-type-options: no
Content-disposition: attac
Expires: Fri, 01 Jan 1990
X-google-cache-control: re
Server: GSE
Via: HTTP/1.1 GWA
Pragma: no-cache
```

- For that you have to add **Authorization header** as well **GData –Version = 3.0** as well.
- When you add **Authorization header** value should be taken from **Google Oath 2.0 Playground** (Access token) and Append “**Bearer+ “ “+{Access token}”**”.
- You can use API URL request as below. For **successful response** you should get **200 OK** status value.

GET ▼
https://www.google.com/m8/feeds/contacts/curtinudara@gmail.com/full

Authorization
Headers (2)
Body
Pre-request Script
Tests

	Key	Value
<input checked="" type="checkbox"/>	Authorization	Bearer ya29.II_BB6wvpcPIBhsU848Af5hZE9wG5TxFB3I8...
<input checked="" type="checkbox"/>	GData-Version	3.0

```
GET https://www.google.com/m8/feeds/contacts/curtinudara@gmail.com/full Params Send
ly Cookies Headers (13) Test Results Status: 200 OK
retty Raw Preview XML
1 <?xml version='1.0' encoding='UTF-8'?>
2 <feed xmlns='http://www.w3.org/2005/Atom' xmlns:gd='http://schemas.google.com/g/2005' xmlns:batch='http://schemas.google.com/gdata/batch' ;
  :gContact='http://schemas.google.com/contact/2008' xmlns:openSearch='http://a9.com/-/spec/opensearch/1.1/' gd:etag='"FVMWRApBDSlgi
  ..&quot;'>
3   <id>curtinudara@gmail.com</id>
4   <updated>2020-03-06T11:11:09.069Z</updated>
5   <category scheme='http://schemas.google.com/g/2005#kind' term='http://schemas.google.com/contact/2008#contact'/>
6   <title>udara curtin's Contacts</title>
7   <link href='http://www.google.com/' rel='alternate' type='text/html'/>
8   <link href='https://www.google.com/m8/feeds/contacts/curtinudara%40gmail.com/full' rel='http://schemas.google.com/g/2005#feed' type
  ='application/atom+xml'/>
9   <link href='https://www.google.com/m8/feeds/contacts/curtinudara%40gmail.com/full' rel='http://schemas.google.com/g/2005#post' type
  ='application/atom+xml'/>
10  <link href='https://www.google.com/m8/feeds/contacts/curtinudara%40gmail.com/full/batch' rel='http://schemas.google.com/g/2005#batch'
  ='application/atom+xml'/>
11  <link href='https://www.google.com/m8/feeds/contacts/curtinudara%40gmail.com/full?max-results=25' rel='self' type='application/atom+xml'
12  <author>
13    <name>udara curtin</name>
14    <email>curtinudara@gmail.com</email>
15  </author>
16  <generator version='1.0' uri='http://www.google.com/m8/feeds'>Contacts</generator>
17  <openSearch:totalResults>1</openSearch:totalResults>
18  <openSearch:startIndex>1</openSearch:startIndex>
19  <openSearch:itemsPerPage>25</openSearch:itemsPerPage>
```

Task 02 (Uses Yodiz API)

- Navigate to URL "<https://app.yodiz.com/user/signup.vz>" and create an account.
- (Save the given [username],[email address],[password] for future use.

Base URL = "https://app.yodiz.com/api/rest/v1/"

Yodiz API Url = <https://www.yodiz.com/api/#introduction>

Authenticate Yodiz API


Send Post request = <https://app.yodiz.com/api/rest/v1/login>



Add the following headers

api-key = 210af5ff-1060-4a4c-98cb-f01bda19afe9

Content-Type = application/json



You will get the API token as the response. Then use that api-token as a header to send the rest of other requests

POST  <https://app.yodiz.com/api/rest/v1/login>

Authorization  Headers (2) Body  Pre-request Script Tests

	Key	Value
<input checked="" type="checkbox"/>	Content-Type	application/json
<input checked="" type="checkbox"/>	api-key	210af5ff-1060-4a4c-98cb-f01bda19afe9
	New key	Value

Body Cookies Headers (6) Test Results

Pretty Raw Preview JSON  


```

1 {
2   "api-token": "HM8OSb5m6l2DJLeWDekjSw"
3 }
```

Then use the following Get request to get all project specific issues

GET Request = <https://app.yodiz.com/api/rest/v1/projects/1/issues?fields=all>

- Use following API keys and tokens as headers to get all issues

GET  <https://app.yodiz.com/api/rest/v1/projects/1/issues?fields=all>

Authorization Headers (3) Body Pre-request Script Tests

	Key	Value
<input checked="" type="checkbox"/>	Content-Type	application/json
<input checked="" type="checkbox"/>	api-key	210af5ff-1060-4a4c-98cb-f01bda19afe9
<input checked="" type="checkbox"/>	api-token	HM8OSb5m6l2DJLeWDekjSw

Send the request and check the response. You should get **JSON response** as below. If you send correct request you will get **200OK status** as follows.

GET ▼ <https://app.yodiz.com/api/rest/v1/projects/1/issues?fields=all>

Body Cookies Headers (7) Test Results

Pretty Raw Preview JSON ▼

```

1 [
2   {
3     "id": 125,
4     "title": "Does not work",
5     "status": {
6       "code": "err-st-30",
7       "narrative": "Resolved"
8     },
9     "severity": {
10      "code": "err-svr-40",
11      "narrative": "Critical"
12    },
13    "release": {
14      "id": 27
15    },
16    "sprint": {
17      "id": 91
18    },
19    "effortEstimate": 0,
20    "effortRemaining": 0,
21    "effortLogged": 0,
22    "highPriority": false,
23    "dueDate": "2017-01-04T00:00:00.000+0000",
24    "mscwPriority": {
25      "code": "err-pr-10"
26    },
27    "tags": [
28      {
29        "id": 113,
30        "title": "graphics",
31        "type": 0
32      }
33    ],
34    "createdBy": {
35

```

Try out providing ticket ID as below.

Send the POST request as follows to create new user story point in the API.

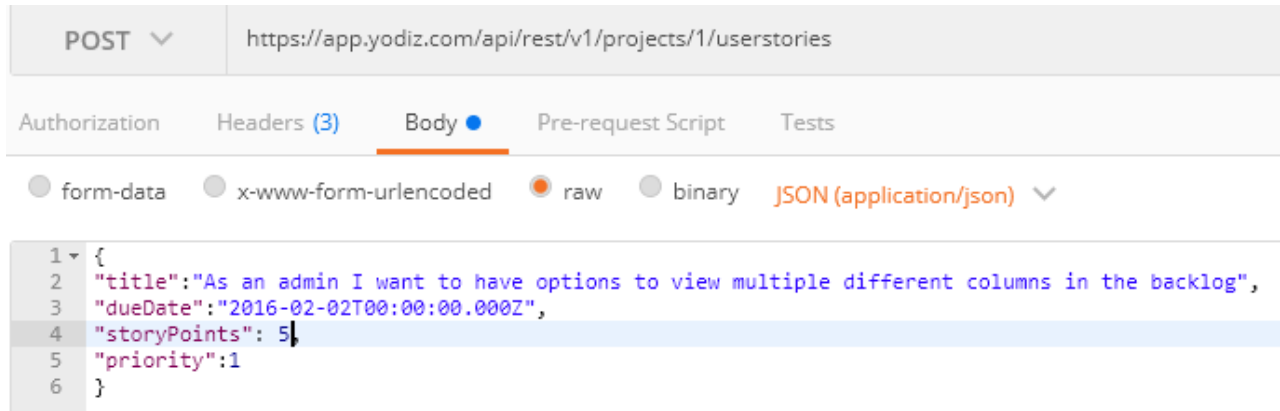
POST ▼ <https://app.yodiz.com/api/rest/v1/projects/1/userstories>

Authorization Headers (3) Body ● Pre-request Script Tests

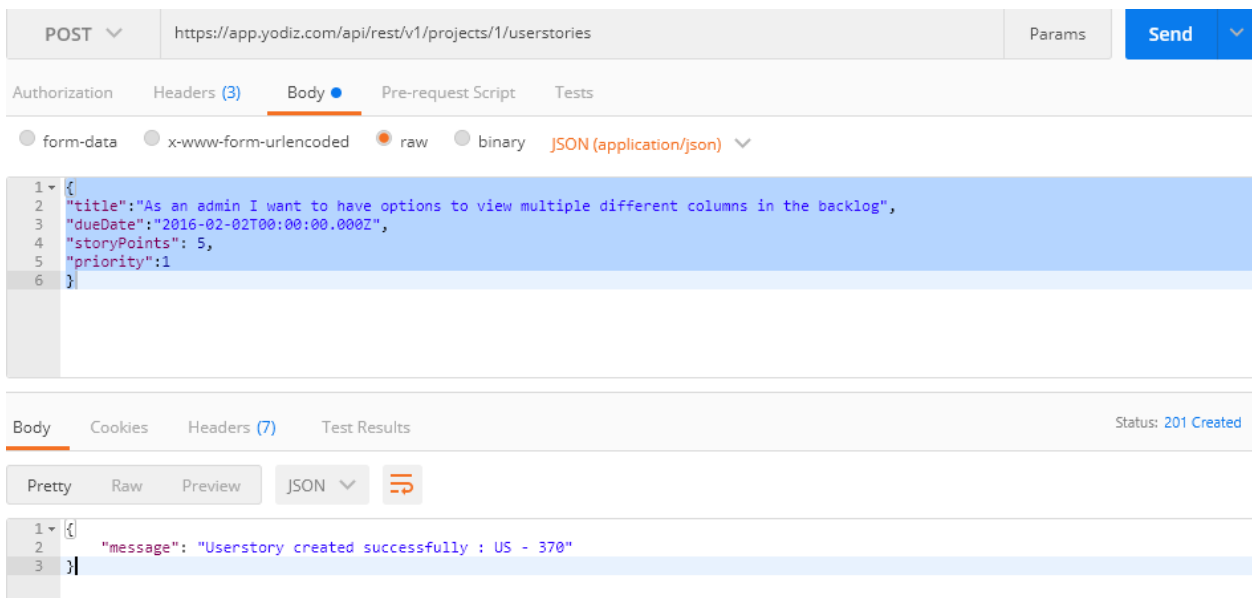
Key	Value
<input checked="" type="checkbox"/> api-token	HM8OSb5m6I2DJLeWDekjSw
<input checked="" type="checkbox"/> api-key	210af5ff-1060-4a4c-98cb-f01bda19afe9
<input checked="" type="checkbox"/> Content-Type	application/json
<input type="text" value="New key"/>	Value

Then in the POST request body you should add following JSON body. In the Postman body select the type raw and try out the following request.

API URL = <https://app.yodiz.com/api/rest/v1/projects/1/userstories>



If it is successful you will get **http status code 201** and response message will be User story created successfully.



Task 03 (Use WSO2 ESB to test SOAP web service)

In this task you should access **3rd party web service** through **WSO2 ESB** and you are going to test the service through **ESB**. For that you are going to access **Calculator Service** hosted in <http://www.dneonline.com/calculator.asmx>. This is a SOAP web service you are going to get wsdl of it.

1) Navigate to `\wso2esb-4.8.0\wso2esb-4.8.0\bin` location and run `wso2server.bat` file. If you are in **Unix** environment run `wso2server.sh` file [`./wso2server.sh`]

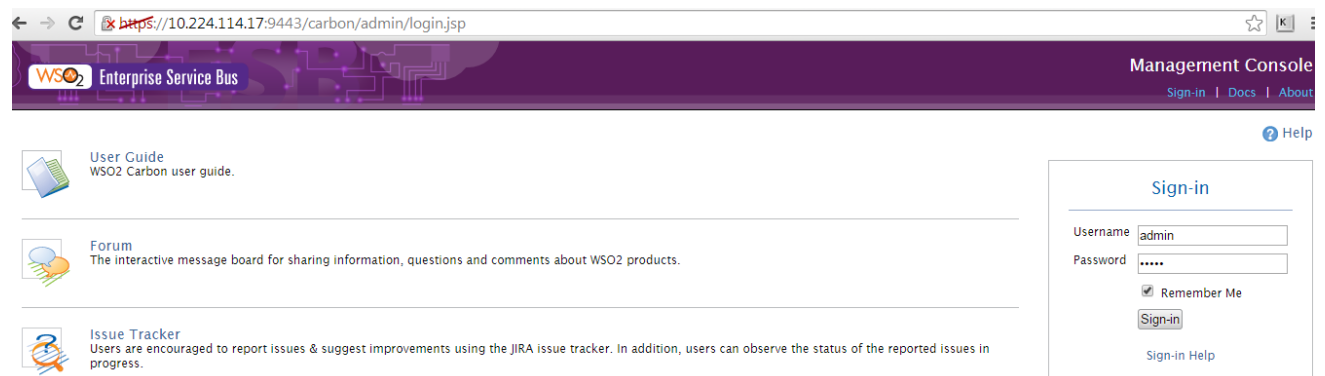
2) Once this is up and running in console it prompts **Management console Url** as below. Copy the **url** and proceed through your browser. [Accept the security exception prompts in browser since **https**] Continue up to **login console**.

```
[2015-01-31 09:46:26.849] INFO - PassThroughHttpSSLListener Starting Pass-through HTTPS Listener...
[2015-01-31 09:46:26.886] INFO - PassThroughHttpSSLListener Pass-through HTTPS Listener started on 0:0:0:0:0:0:8243
[2015-01-31 09:46:26.886] INFO - PassThroughHttpListener Starting Pass-through HTTP Listener...
[2015-01-31 09:46:26.984] INFO - PassThroughHttpListener Pass-through HTTP Listener started on 0:0:0:0:0:0:8280
[2015-01-31 09:46:26.952] INFO - NioSelectorPool Using a shared selector for servlet write/read
[2015-01-31 09:46:27.492] INFO - NioSelectorPool Using a shared selector for servlet write/read
[2015-01-31 09:46:27.531] INFO - RegistryEventingServiceComponent Successfully Initialized Eventing on Registry
[2015-01-31 09:46:28.206] INFO - JMXServerManager JMX Service URL : service:jmx:rmi://localhost:1111/jndi/rmi://localhost:9999/jmxrmi
[2015-01-31 09:46:28.206] INFO - StartupFinalizerServiceComponent Server : WSO2 Enterprise Service Bus-4.8.0
[2015-01-31 09:46:28.207] INFO - StartupFinalizerServiceComponent WSO2 Carbon started in 88 sec
[2015-01-31 09:46:28.612] INFO - CarbonUIServiceComponent Mgt Console URL : https://10.224.114.17:9443/carbon/
[2015-01-31 10:15:08.542] INFO - CarbonAuthenticationUtil 'admin@carbon.super [-1234]' logged in at [2015-01-31 10:15:08.537+0530]
[2015-01-31 10:34:57.218] INFO - CarbonAuthenticationUtil 'admin@carbon.super [-1234]' logged in at [2015-01-31 10:34:57.218+0530]
```

3) WSO2 ESB management console login credentials are as below. Login to Management console using them

Username = admin

Password = admin



4) Now you are going to access **3rd party web service** already hosted in remotely. For that **Navigate to URL =** <http://www.dneonline.com/calculator.asmx?WSDL>

5) Go to view page source and copy whole WSDL content and save it as **Calculator.wsdl** in `wso2esb-4.8.0\wso2esb-4.8.0\repository\samples\resources\proxy` location of your **ESB**. This **wsdl** file would act as a **proxy service** for **ESB** to connect **remote service**.

Create Proxy service (WSDL based proxy)

6) In management console click **Proxy service** => **WSDL based proxy**. Then you have to fill following **Proxy service settings details**. Click Test URI button and check the service availability. Refer below screenshots.

Home > Manage > Services > Add > Proxy Service

Create Proxy Service from Template

Create a new proxy service. Select a template that match

Select Template	
Pass Through Proxy	Create a simple proxy through the proxy.
Secure Proxy	Create a proxy with backend service.
WSDL Based Proxy	Create a proxy service from an actual service.
Logging Proxy	Create a proxy service from the backend service.
Transformer Proxy	Create a proxy service that transforms responses coming from the backend service.
Custom Proxy	Launch the proxy service including sequence.

7) Once details are given click **Create Button**.

WSDL Based Proxy

Create a proxy service out of a WSDL of an existing Web Service. Endpoint information is extracted from a remotely hosted WSDL of an actual service

Proxy Service Settings

Proxy Service Name*

WSDL URI*

WSDL Service*

WSDL Port*

Publish Same Service Contract ☒

☐ Service Contract Publication Options

☐ Transports

8) Once you created proxy service you should be able to see it under list of proxies. Then click **List** under services tab in main menu.

WSO2 Enterprise Service Bus Management Console

Signed-in as: admin@carbon.super | Sign-out | Docs

Home > Manage > Services > List

Deployed Services

5 active services. 5 deployed service group(s).

Service Type: ALL Service:

Select all in this page | Select none [Delete](#)

Services	Proxy	Security	WSDL1.1	WSDL2.0	Try this service	Download	Design View	Source View
<input type="checkbox"/> Calculator.wSDL	proxy	Unsecured	WSDL1.1	WSDL2.0	Try this service	Download	Design View	Source View
<input type="checkbox"/> echo	axis2	Unsecured	WSDL1.1	WSDL2.0	Try this service	Download	Design View	Source View
<input type="checkbox"/> GoogleContacts	axis2	Unsecured	WSDL1.1	WSDL2.0	Try this service	Download	Design View	Source View
<input type="checkbox"/> Version	axis2	Unsecured	WSDL1.1	WSDL2.0	Try this service	Download	Design View	Source View
wso2carbon-sts	sts	Unsecured	WSDL1.1	WSDL2.0				

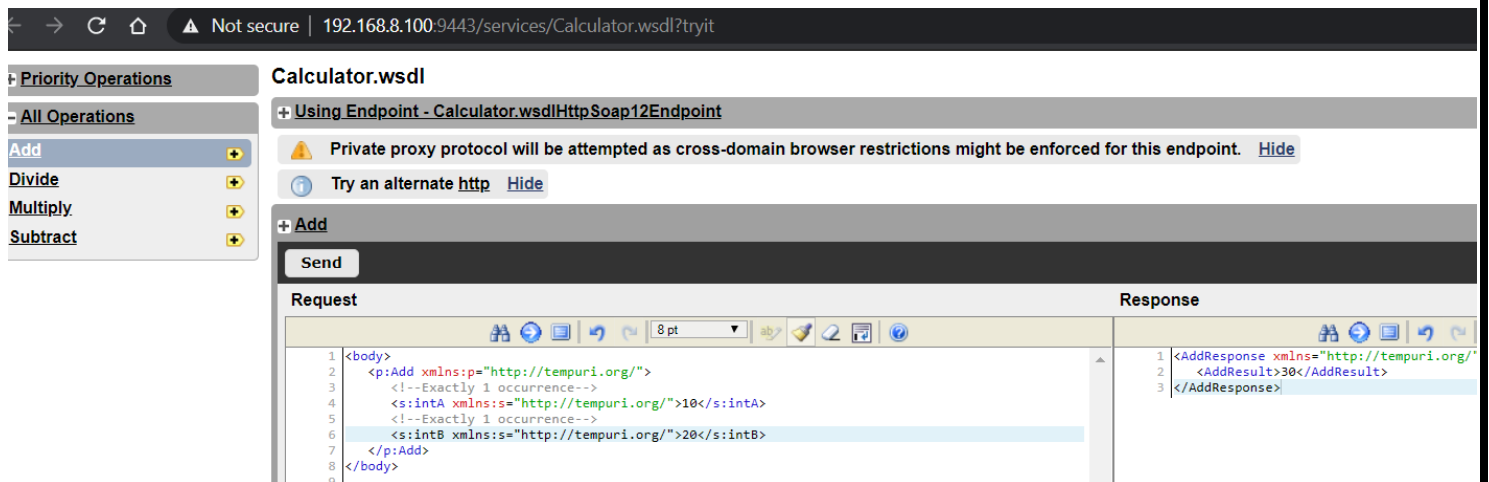
9) It generates proxy service as follows to your remote service

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <proxy xmlns="http://ws.apache.org/ns/synapse"
3     name="Calculator.wSDL"
4     startOnLoad="true"
5     statistics="disable"
6     trace="disable"
7     transports="https,http">
8   <target>
9     <outSequence>
10       <send/>
11     </outSequence>
12     <endpoint>
13       <wsdl port="CalculatorSoap"
14           service="Calculator"
15           uri="http://www.dneonline.com/calculator.asmx?WSDL"/>
16     </endpoint>
17   </target>
18   <publishWSDL uri="http://www.dneonline.com/calculator.asmx?WSDL"/>
19   <description/>
20 </proxy>

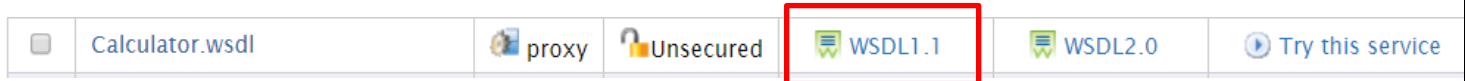
```

10) In list of proxy services select and click **Try This service**. Now modify SOAP request with adding values and monitor the response. You will get relevant response.

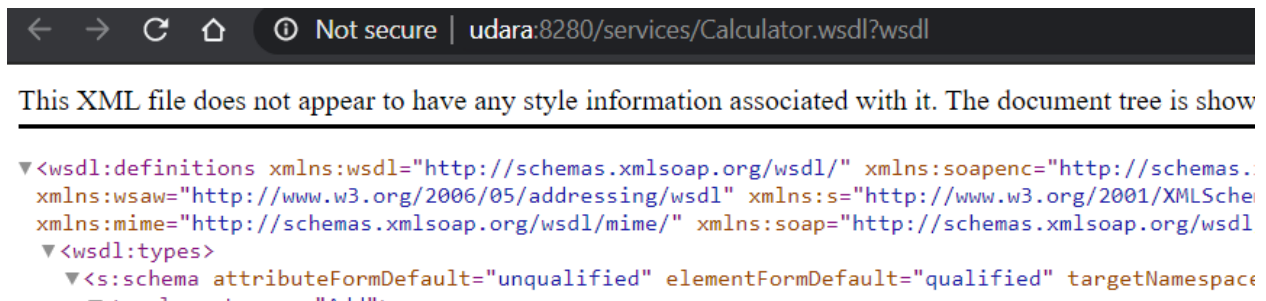


Task 04 (Use SOAP UI to send request for ESB)

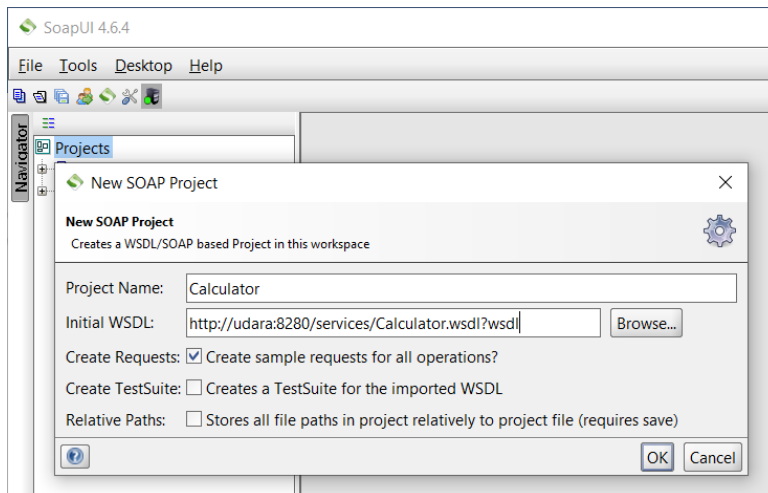
- 1) Open SOAP UI version 4.5.X. Then click **soapui.bat** file (in **Windows**) or run **./soapui.sh** for **UNIX environment**. File located = **soapui-4.5.1-win32-standalone-bin\soapui-4.5.1\bin**
- 2) Now in **WSO2 ESB management console** select the relevant proxy service and click **WSDL.1.1** as below.



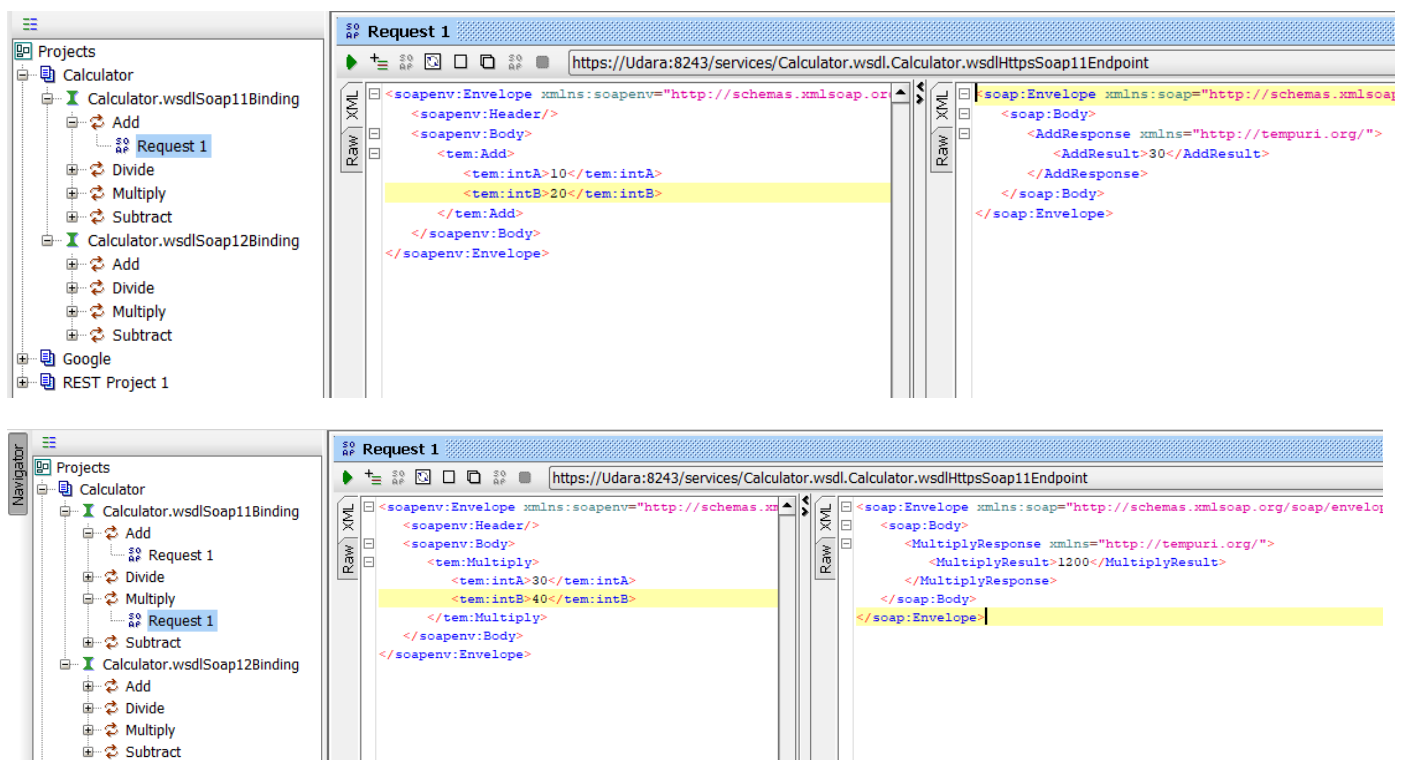
- 3) Then copy the **WSDL URL** in browser. This is the service for you are going to access to connect Proxy service. Now **SOAP UI** sends the request to **ESB** and **ESB** sends the request to End point.



- 4) Copy the WSDL url by selecting WSDL1.1 in management console of ESB. Then in SOP UI, select **File => New SOAP UI project** and paste URL as below.



5) It automatically get project name as Temperature and create SOAP project as below. Then in the project **select request1** and **double click**. SOAP request automatically get generated. Modify the request with suitable values. You will get below response.



The End