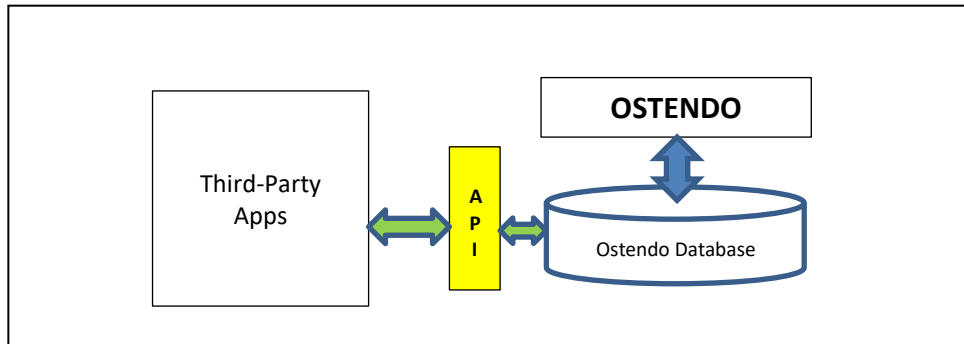


OSTENDO API SERVICE

The Ostendo API Rest Service provides the facility for external third-party applications to integrate with Ostendo. Any application which can make HTTP calls can integrate with Ostendo via this service. Even common applications like MS Excel can be integrated with Ostendo via user-written macros which talks to the Ostendo API Service.



When the Ostendo API Service is switched on or STARTED, external applications can GET information from or POST information to any table in the Ostendo database. A unique feature of the Ostendo API service is the ability to post SQL queries to extract specific data from one or more tables.

Please note that even if all users are logged off from Ostendo, the API service remains active and running until it is STOPPED.

The data format used is primarily XML.

As a security measure, only developers/applications which are defined and authorised in the Ostendo database can have access to the API Service. It is recommended that, for enhanced security, use HTTPS instead of HTTP.

This document covers the following areas:

- the basic HTTP(S) Methods and Resources of the Ostendo API,
- how to configure the API service
- how to authorize API users and generate the API key
- how to start and stop the Ostendo API service.

Note to Third-Party Developers:

This document is written primarily for Ostendo Consultants and Users. If you are new to Ostendo, you may want to work with an experienced Ostendo Consultant to build your integration solution to take advantage of the functional richness of the Ostendo ERP system.

You can get a report listing all the tables in the Ostendo database by going to the **General menu – Reports – Full Listing of Tables.**

This report allows you to select one or more tables to be printed or previewed on screen. The report lists all the fieldnames, field types, length, and any Required fields in each table.

This report is useful if your application needs to query or update the table.

INTRODUCTION

The general Ostendo API URL structure looks like this:

http://ipaddress:port/resource?apikey=URLEncoded_APIKey&configuration=0

or

https://domainname:port/resource?apikey=URLEncoded_APIKey&configuration=0

For HTTP, if the Ostendo application is sitting in the same machine as the calling application, then the **ipaddress** can be represented by "**localhost**". If the Ostendo application is sitting on a server machine, then the **ipaddress of the server** needs to be specified.

For HTTPS, use SSL certified **domain name** (instead of ip address), and the allocated https port.

If the third-party application is accessing the Ostendo API via the internet, then you may need to have a **fixed or static IP address**.

The **Port** address is the port specified when setting up the Ostendo API Configuration. This port should be **dedicated** to Ostendo API use.

The Ostendo API **resource** could be as general as "tabledata" or specific like "salesorder", "joborder", "purchaseorder", etc..

A **URL-encoded API Key** is generated by Ostendo for each third-party application authorised by that Ostendo installation. API calls will not be processed without presenting a valid URL-encoded API Key.

The Ostendo API Service allows up to 10 "**configurations**" to be defined. Each configuration could refer to a different Ostendo Database. This is useful for specifying one configuration for testing or development purposes and another for access to the LIVE database, etc.

The configuration parameter is **optional**. Values range from "0" to "9". If not specified, the **default value is "0"**.

HTTP(S) Methods supported:

GET - To retrieve a specified record or table data.

POST - To create or add records to specified resources.

DELETE - To delete a specified record or tabledata.

Response / Status Codes:

200 OK - Request is accepted and executed.

400 Bad Request - Syntax error in the request.

401 Unauthorized - Either the Database Configuration is not available or the API Key is invalid.

500 Internal Server Error - Server error or exception

501 Not Implemented - Resource not recognized by Ostendo API Service

A. List of Resources using the GET method

Using the HTTP or HTTPS GET method, you can have access to various Ostendo API resources. These resources allow the calling application to retrieve data from various tables in the Ostendo database.

1. salesorder/[ordernumber]

- Specifying this resource in your GET call will retrieve the specified Sales Order header and associated lines from the Ostendo database. You can get only one order per call.
- Output Format: XML only

Sample URL:

<http://localhost:82/salesorder/SO300017?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/salesorder/SO300017?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

XML returned:

```
<?xml version="1.0" encoding="utf-8"?>
<salesorder>
  <orderheader>
    <ordernumber>SO300017</ordernumber>
    <orderstatus>Open</orderstatus>
    <orderdate>25/07/2011</orderdate>
    <requireddate>25/07/2011</requireddate>
    <ordertype>CounterSales</ordertype>
    <orderstyle>Counter</orderstyle>
    <printstatus>Printed</printstatus>
    <customer>Jim Gold & amp; Co Ltd</customer>
    <orderaddress1>11 Wild Berry Cove</orderaddress1>
    <orderaddress2>Boolanga</orderaddress2>
    <orderpostalcode>4500</orderpostalcode>
    <orderstate>QLD</orderstate>
    <ordercity>Queensland</ordercity>
    <ordercountry>Australia</ordercountry>
    <ordercontact>Ken Jolly</ordercontact>
    <orderphone>44 312 587 6544</orderphone>
    <orderfax>44 312 587 6545</orderfax>
    <orderemail>ken@development-x.com</orderemail>
    <billingcustomer>Jim Gold & amp; Co Ltd</billingcustomer>
    
    <iineunitarea>u</iineunitarea>
    <linetotalweight>0</linetotalweight>
    <linetotalvolume>0</linetotalvolume>
    <linetotalarea>0</linetotalarea>
    <rentalline>False</rentalline>
    <rentalreturned>False</rentalreturned>
    <calculatedunitprice>16.45</calculatedunitprice>
    <packsizeqty>5</packsizeqty>
    <onesteppickqty>5</onesteppickqty>
    <onesteppreviousqty>0</onesteppreviousqty>
    <onesteppreviousoldqty>0</onesteppreviousoldqty>
    <orderdiscountapplies>False</orderdiscountapplies>
    <sysdatecreated>25/07/2011 11:07:54 a.m.</sysdatecreated>
    <sysdatemodified>25/07/2011 11:07:54 a.m.</sysdatemodified>
    <sysuniqueid>116697</sysuniqueid>
    <sysusercreated>ADMIN</sysusercreated>
  </orderline>
</orderheader>
</salesorder>
```

2. **joborder/[ordernumber]**

- Specifying this resource in your GET call will retrieve a Job Order header and associated lines from the Ostendo database. You can get only one order per call.
- Output Format: XML only

Sample URL:

<http://localhost:82/joborder/JOB400004?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/joborder/JOB400004?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

3. **purchaseorder/[ordernumber]**

- Specifying this resource in your GET call will retrieve a Purchase Order header and associated lines from the Ostendo database. You can get only one order per call.
- Output Format: XML only

Sample URL:

<http://localhost:82/purchaseorder/PO100013?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/purchaseorder/PO100013?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

4. **assemblyorder/[ordernumber]**

- Specifying this resource in your GET call will retrieve a Assembly Order header and associated lines from the Ostendo database. You can get only one order per call.
- Output Format: XML only

Sample URL:

<http://localhost:82/assemblyorder/WO200011?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/assemblyorder/WO200011?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

5. **salesinvoice/[invoicenumbr]**

- Specifying this resource in your GET call will retrieve a SalesInvoice header and associated lines from the Ostendo database. You can get only one invoice per call.
- Output Format: XML only

Sample URL:

<http://localhost:82/salesinvoice/500073?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/salesinvoice/500073?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

6. tabledata

- Specifying this resource in your GET call will retrieve records from a specified Ostendo database table.
- Conditions can be set to filter the required records (e.g. **condition=fieldname1=fieldvalue1 and fieldname2>=fieldvalue2** , where fieldvalues must be enclosed in single quotes (%27 in URL-encoded form) if they are string values.
- Output Format: XML or JSON (must specify a format)
- Optional parameter : includeblankvalues= (true or false)

Sample URL: (no conditions)

<http://localhost:82/tabledata?tablename=standardunits&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=xml&configuration=0>

<https://domainname:port/tabledata?tablename=standardunits&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=xml&configuration=0>

Sample URL: (with conditions. The condition value must be URL-encoded.)

<http://localhost:82/tabledata?tablename=standardunits&condition=standardunit > %27Each%27 and timeperhour > 1&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=json&configuration=0>

<https://domainname:port/tabledata?tablename=standardunits&condition=standardunit > %27Each%27 and timeperhour > 1&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=json&configuration=0>

Results:

```
[
  {
    "STANDARDUNIT":"Hours",
    "UNITDESCRIPTION":"Hours",
    "ISTIMEUNIT":1,
    "TIMEPERHOUR":1,
    "UPPERSTANDARDUNIT":"HOURS",
    "SYSDATECREATED":"14/11/2006 1:54:36 p.m.",
    "SYSDATEMODIFIED":"14/11/2006 1:54:36 p.m.",
    "SYSUSERCREATED":"",
    "SYSUSERMODIFIED":""
  },
  {
    "STANDARDUNIT":"Minutes",
    "UNITDESCRIPTION":"Minutes",
    "ISTIMEUNIT":1,
    "TIMEPERHOUR":60,
    "UPPERSTANDARDUNIT":"MINUTES",
    "SYSDATECREATED":"7/12/2007 1:00:02 p.m.",
    "SYSDATEMODIFIED":"7/12/2007 1:00:02 p.m.",
    "SYSUSERCREATED":"ADMIN",
    "SYSUSERMODIFIED":""
  },
  {
    "STANDARDUNIT":"Half Hour",
    "UNITDESCRIPTION":"30 Minutes",
    "ISTIMEUNIT":1,
    "TIMEPERHOUR":2,
    "UPPERSTANDARDUNIT":"HALF HOUR",
    "SYSDATECREATED":"7/12/2007 1:00:23 p.m.",
    "SYSDATEMODIFIED":"7/12/2007 1:00:23 p.m.",
    "SYSUSERCREATED":"ADMIN",
    "SYSUSERMODIFIED":""
  },
  {
    "STANDARDUNIT":"Quarter Hour",
    "UNITDESCRIPTION":"15 Minutes",
    "ISTIMEUNIT":1,
    "TIMEPERHOUR":4,
    "UPPERSTANDARDUNIT":"QUARTER HOUR",
    "SYSDATECREATED":"7/12/2007 1:00:41 p.m.",
    "SYSDATEMODIFIED":"7/12/2007 1:00:41 p.m.",
  }
```

```
"SYSUSERCREATED":"ADMIN",  
"SYSUSERMODIFIED":""  
}  
]
```

B. List of Resources using the POST method

Using the HTTP or HTTPS POST method, you can send data to various Ostendo API resources. These resources allow the calling application to add records to the specified tables in the Ostendo database.

1. Salesorder

- Specifying this resource in your POST call will send the sales order data into Ostendo and create the Sales Orders header and lines records.
- Content: Sales order data in XML format only. Content can include one or more orders.
If order numbering is set to "automatic" in Ostendo, then the ordernumber field must not contain any value.

Sample Content: (Lines in bold are mandatory)

```
<?xml version="1.0" encoding="utf-8"?>
<salesorder>
  <orderheader>
    <ordernumber></ordernumber>
    <orderdate>26/03/2014</orderdate>
    <ordertype>DeliveryOrder</ordertype>
    <orderdescription>Office Desk</orderdescription>
    <customer>Jim Gold & Co Ltd</customer>
    <orderaddress1>11 Wild Berry Cove</orderaddress1>
    <orderaddress2>Boolanga</orderaddress2>
    <orderpostalcode>4500</orderpostalcode>
    <orderstate>QLD</orderstate>
    <ordercity>Queensland</ordercity>
    <ordercountry>Australia</ordercountry>
    <billingcustomer>Jim Gold & Co Ltd</billingcustomer>
    <billingaddress1>Box 45-234</billingaddress1>
    <billingaddress2>Boolanga</billingaddress2>
    <billingpostalcode>4500</billingpostalcode>
    <billingstate>QLD</billingstate>
    <billingcity>Queensland</billingcity>
    <billingcountry>Australia</billingcountry>
    <taxgroup>TAXABLE</taxgroup>
    <creditterm>20th of Month</creditterm>
    <orderline>
      <ordernumber></ordernumber>
      <linenumber>10</linenumber>
      <codetype>Item Code</codetype>
      <linecode>OD-7001</linecode>
      <linedescription>Pine Office Desk 2180mm x 600mm with 2 Drawers</linedescription>
      <lineunit>Each</lineunit>
      <orderqty>1</orderqty>
      <orderunitprice>261.1</orderunitprice>
      <orderunittax>32.6375</orderunittax>
      <orderunitinclprice>0</orderunitinclprice>
      <discountpercent>0</discountpercent>
      <discountamount>0</discountamount>
      <extendednettprice>261.1</extendednettprice>
      <extendedtax>32.64</extendedtax>
      <extendedtotalprice>293.74</extendedtotalprice>
      <taxcode>GST</taxcode>
      <priceoverride>False</priceoverride>
      <standardunitprice>261.1</standardunitprice>
      <customerunitprice>261.1</customerunitprice>
    </orderline>
  </orderheader>
  <orderheader>
    <ordernumber></ordernumber>
    <orderdate>27/03/2014</orderdate>
    <ordertype>DeliveryOrder</ordertype>
    <orderdescription>Office Chairs</orderdescription>
    <customer>Green Fingers Maloy Ltd</customer>
    <orderaddress1>72 Mower Place</orderaddress1>
    <orderaddress2>Uxbridge Square</orderaddress2>
    <orderpostalcode>3250</orderpostalcode>
    <orderstate>VIC</orderstate>
    <ordercity>Melbourne</ordercity>
    <ordercountry>Australia</ordercountry>
    <billingcustomer>Green Fingers Maloy Ltd</billingcustomer>
    <billingaddress1>Private Bag</billingaddress1>
```

```

<billingaddress2>Uxbridge Square</billingaddress2>
<billingpostalcode>3250</billingpostalcode>
<billingstate>VIC</billingstate>
<billingcity>Melbourne</billingcity>
<billingcountry>Australia</billingcountry>
<taxgroup>TAXABLE</taxgroup>
<creditterm>20th of Month</creditterm>
  <orderline>
    <ordernumber></ordernumber>
    <linenumber>10</linenumber>
    <codetype>Item Code</codetype>
    <linecode>OC-7450</linecode>
    <linedescription>Office Chair - Black Leather Executive</linedescription>
    <lineunit>Each</lineunit>
    <orderqty>2</orderqty>
    <orderunitprice>269.1</orderunitprice>
    <orderunittax>33.6375</orderunittax>
    <orderunitinclprice>0</orderunitinclprice>
    <discountpercent>10</discountpercent>
    <discountamount>59.8</discountamount>
    <extendednetprice>538.2</extendednetprice>
    <extendedtax>67.28</extendedtax>
    <extendedtotalprice>605.48</extendedtotalprice>
    <taxcode>GST</taxcode>
    <priceoverride>True</priceoverride>
    <standardunitprice>299</standardunitprice>
    <customerunitprice>299</customerunitprice>
  </orderline>
</orderheader>
</salesorder>

```

Sample URL:

<http://localhost:82/salesorder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/salesorder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

Results: The Responsevalue contains the Sales OrderNumber(s) created.

```

<?xml version="1.0" encoding="utf-8"?>
  <ostendoapi>
    <ostendoapiversion>1.0.0.88</ostendoapiversion>
    <responsestatus>ok</responsestatus>
    <responsevalue>DO300166,DO300167</responsevalue>
  </ostendoapi>

```


2. Joborder

- Specifying this resource in your POST call will send the job order data into Ostendo and create the Job Orders header and lines records.
- Content: Job order data in XML format only. Content can include one or more orders.
If order numbering is set to "automatic" in Ostendo, then the ordernumber field must not contain any value.

Sample Content: (Lines in bold are mandatory)

```
<?xml version="1.0" encoding="utf-8"?>
<joborder>
  <orderheader>
    <ordernumber></ordernumber>
    <orderdate>7/04/2014</orderdate>
    <requireddate>7/04/2014</requireddate>
    <orderdescription>ComputerNetworking</orderdescription>
    <jobtype>Service</jobtype>
    <customer>Marvellous Company (Pty) Ltd</customer>
    <orderaddress1>11 Gold Street</orderaddress1>
    <orderaddress2>New Brighton</orderaddress2>
    <orderpostalcode>8083</orderpostalcode>
    <orderstate>SI</orderstate>
    <ordercity>Christchurch</ordercity>
    <ordercountry>New Zealand</ordercountry>
    <orderphone>03 234 5678</orderphone>
    <orderfax>03 234 5679</orderfax>
    <orderemail>mark@marvellous.co.nz</orderemail>
    <billingcustomer>Marvellous Company (Pty) Ltd</billingcustomer>
    <billingaddress1>PO Box 111333</billingaddress1>
    <billingaddress2>New Brighton</billingaddress2>
    <billingpostalcode>8083</billingpostalcode>
    <billingstate>SI</billingstate>
    <billingcity>Christchurch</billingcity>
    <billingcountry>New Zealand</billingcountry>
    <billingphone>03 234 5678</billingphone>
    <billingfax>03 234 5679</billingfax>
    <billingemail>mark@marvellous.co.nz</billingemail>
    <taxgroup>TAXABLE</taxgroup>
    <creditterm>20th of Month</creditterm>
    <orderline>
      <ordernumber></ordernumber>
      <codetype>Task Bill Code</codetype>
      <linecode>PCTONETWORK</linecode>
      <linedescription>Task Bill for connecting a PC to a Local Area Network</linedescription>
      <lineunit>Each</lineunit>
      <orderqty>1</orderqty>
      <orderunitprice>56.49</orderunitprice>
      <orderunittax>5.649</orderunittax>
      <orderunitinclprice>62.14</orderunitinclprice>
      <discountpercent>0</discountpercent>
      <discountamount>0</discountamount>
      <extendednettpri<strong>ce>56.49</strong>/extendednettpri<strong>ce>
      <extendedtax>5.65</extendedtax>
      <extendedtotalprice>62.14</extendedtotalprice>
      <taxcode>GST</taxcode>
      <taskname>NetworkConnect</taskname>
      <standardunitprice>56.49</standardunitprice>
      <customerunitprice>56.49</customerunitprice>
      <priceoverride>False</priceoverride>
    </orderline>
    <orderline>
      <ordernumber></ordernumber>
      <codetype>Task Bill Code</codetype>
      <linecode>PCANITVIRUSINSTALL</linecode>
      <linedescription>Install Antivirus Software on PC</linedescription>
      <lineunit>Each</lineunit>
      <orderqty>1</orderqty>
      <orderunitprice>137.9</orderunitprice>
      <orderunittax>13.79</orderunittax>
      <orderunitinclprice>151.69</orderunitinclprice>
      <discountpercent>0</discountpercent>
      <discountamount>0</discountamount>
      <extendednettpri<strong>ce>137.9</strong>/extendednettpri<strong>ce>
      <extendedtax>13.79</extendedtax>
      <extendedtotalprice>151.69</extendedtotalprice>
    </orderline>
  </orderheader>
</joborder>
```

```

        <taxcode>GST</taxcode>
        <taskname>AntivirusInstall</taskname>
        <standardunitprice>137.9</standardunitprice>
        <customerunitprice>137.9</customerunitprice>
        <priceoverride>False</priceoverride>
    </orderline>
</orderline>
        <ordernumber></ordernumber>
        <codetype>Labour Code</codetype>
        <linecode>LAB-SERVICE</linecode>
        <linedescription>On-Site Service Labour</linedescription>
        <lineunit>Hours</lineunit>
        <orderqty>2</orderqty>
        <orderunitprice>75</orderunitprice>
        <orderunittax>7.5</orderunittax>
        <orderunitinclprice>82.5</orderunitinclprice>
        <discountpercent>0</discountpercent>
        <discountamount>0</discountamount>
        <extendednetprice>150</extendednetprice>
        <extendedtax>15</extendedtax>
        <extendedtotalprice>165</extendedtotalprice>
        <taxcode>GST</taxcode>
        <taskname>BackupRestore</taskname>
        <standardunitprice>75</standardunitprice>
        <customerunitprice>75</customerunitprice>
        <priceoverride>False</priceoverride>
    </orderline>
</orderline>
        <ordernumber>SER400006</ordernumber>
        <linenumber>10</linenumber>
        <codetype>Labour Code</codetype>
        <linecode>LAB-SERVICE</linecode>
        <linedescription>On-Site Service Labour</linedescription>
        <lineunit>Hours</lineunit>
        <orderqty>2.5</orderqty>
        <orderunitprice>75</orderunitprice>
        <orderunittax>7.5</orderunittax>
        <orderunitinclprice>82.5</orderunitinclprice>
        <discountpercent>0</discountpercent>
        <discountamount>0</discountamount>
        <extendednetprice>187.5</extendednetprice>
        <extendedtax>18.75</extendedtax>
        <extendedtotalprice>206.25</extendedtotalprice>
        <taxcode>GST</taxcode>
        <taskname>WindowsTraining</taskname>
        <standardunitprice>75</standardunitprice>
        <customerunitprice>75</customerunitprice>
        <priceoverride>False</priceoverride>
    </orderline>
</orderheader>
</joborder>

```

Sample URL:

<http://localhost:82/joborder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/joborder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

Results: The Responsevalue contains the Job OrderNumber(s) created.

```

<?xml version="1.0" encoding="utf-8"?>
<ostendoapi>
  <ostendoapiversion>1.0.0.88</ostendoapiversion>
  <responsestatus>ok</responsestatus>
  <responsevalue>SER400071</responsevalue>
</ostendoapi>

```

3. Purchaseorder

- Specifying this resource in your POST call will send the purchase order data into Ostendo and create the Purchase Orders header and lines records.
- Content: Purchase order data in XML format only. Content can include one or more orders. If order numbering is set to "automatic" in Ostendo, then the ordernumber field must not contain any value.

Sample Content: (Lines in bold are mandatory)

```
<?xml version="1.0" encoding="utf-8"?>
<purchaseorder>
  <orderheader>
    <ordernumber></ordernumber>
    <orderdate>9/04/2014</orderdate>
    <ordertype>Standard</ordertype>
    <supplier>Camelia Car Co Ltd</supplier>
    <orderaddress1>P O Box 37-400</orderaddress1>
    <orderaddress2>North Shore Mail Centre</orderaddress2>
    <orderpostalcode>1200</orderpostalcode>
    <orderstate>NI</orderstate>
    <ordercity>North Shore City</ordercity>
    <ordercountry>New Zealand</ordercountry>
    <orderphone>443-9999</orderphone>
    <orderfax>443-8888</orderfax>
    <orderemail>info@cameliacar.co.nz</orderemail>
    <deliverto>Company</deliverto>
    <deliveryname>Company</deliveryname>
    <deliveryaddress1>4 Pacific Rise</deliveryaddress1>
    <deliveryaddress2>Mt Wellington</deliveryaddress2>
    <deliverycity>Auckland</deliverycity>
    <deliverycountry>New Zealand</deliverycountry>
    <deliveryphone>+64-9-5253612</deliveryphone>
    <deliveryfax>+64-9-5253614</deliveryfax>
    <taxgroup>TAXABLE</taxgroup>
    <creditterm>20th of Month</creditterm>
    <orderline>
      <ordernumber></ordernumber>
      <linenumber>10</linenumber>
      <codetype>Descriptor Code</codetype>
      <linecode>MATERIAL</linecode>
      <linedescription>Material Used in Progress Claim</linedescription>
      <lineunit>$</lineunit>
      <orderqty>1</orderqty>
      <orderunitprice>10</orderunitprice>
      <orderunittax>1</orderunittax>
      <extendedorderprice>10</extendedorderprice>
      <extendedordertax>1</extendedordertax>
      <priceoverride>True</priceoverride>
      <taxcode>GST</taxcode>
    </orderline>
  </orderheader>
</purchaseorder>
```

Sample URL:

<http://localhost:82/purchaseorder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/purchaseorder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

Results: The Responsevalue contains the Purchase OrderNumber(s) created.

```
<?xml version="1.0" encoding="utf-8"?>
<ostendoapi>
  <ostendoapiversion>1.0.0.88</ostendoapiversion>
  <responsestatus>ok</responsestatus>
  <responsevalue>PO100029</responsevalue>
</ostendoapi>
```

4. Assemblyorder

- Specifying this resource in your POST call will send the assembly order data into Ostendo and create the Assembly Orders header and lines records.
- Content: Assembly order data in XML format only. Content can include one or more orders. If order numbering is set to "automatic" in Ostendo, then the ordernumber field must not contain any value.

Sample Content: (Lines in bold are mandatory)

```
<?xml version="1.0" encoding="utf-8"?>
<assemblyorder>
  <orderheader>
    <ordernumber></ordernumber>
    <orderdate>9/04/2014</orderdate>
    <itemcode>1105-2184</itemcode>
    <itemdescription>Handle Assembly</itemdescription>
    <itemunit>Each</itemunit>
    <requireddate>10/04/2014</requireddate>
    <orderqty>30</orderqty>
    <orderline>
      <ordernumber></ordernumber>
      <stepname>Assembly</stepname>
      <codetype>Item Code</codetype>
      <linecode>760-2176</linecode>
      <linedescription>Tube-Stainless Steel-25mm1200mm</linedescription>
      <lineunit>Each</lineunit>
      <orderqty>30</orderqty>
    </orderline>
    <orderline>
      <ordernumber></ordernumber>
      <stepname>Assembly</stepname>
      <codetype>Labour Code</codetype>
      <linecode>LAB-ASSEMBLY</linecode>
      <linedescription>Assembly Labour</linedescription>
      <lineunit>Hours</lineunit>
      <orderqty>30</orderqty>
    </orderline>
    <orderline>
      <ordernumber>WO200011</ordernumber>
      <stepname>Assembly</stepname>
      <codetype>Item Code</codetype>
      <linecode>900-2182</linecode>
      <linedescription>Handle Grip-Rubber-25mm</linedescription>
      <lineunit>Each</lineunit>
      <orderqty>60</orderqty>
    </orderline>
  </orderheader>
</assemblyorder>
```

Sample URL:

<http://localhost:82/assemblyorder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/assemblyorder?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

Results: The Responsevalue contains the Assembly OrderNumber(s) created.

```
<?xml version="1.0" encoding="utf-8" ?>
<ostendoapi>
  <ostendoapiversion>1.0.0.88</ostendoapiversion>
  <responsestatus>ok</responsestatus>
  <responsevalue>WO200033</responsevalue>
</ostendoapi>
```

5. SalesInvoice

- Specifying this resource in your POST call will send the “Direct” sales invoice data into Ostendo and create the Sales Invoice header and lines records.
- Content: Direct Sales Invoice data in XML format only. Content can include one or more invoices. If Direct Invoice numbering is set to “automatic” in Ostendo, then the ordernumber field must not contain any value.

Sample Content: (Lines in bold are mandatory)

```
<?xml version="1.0" encoding="utf-8"?>
<salesinvoice>
  <invoiceheader>
    <invoicenum></invoicenum>
    <invoicedate>9/04/2014</invoicedate>
    <customer>Jim Gold & Co Ltd</customer>
    <billingaddress1>Box 45-234</billingaddress1>
    <billingaddress2>Boolanga</billingaddress2>
    <billingpostalcode>4500</billingpostalcode>
    <billingstate>QLD</billingstate>
    <billingcity>Queensland</billingcity>
    <billingcountry>Australia</billingcountry>
    <billingphone>61-7-434-5618</billingphone>
    <billingfax>61-7-434-5619</billingfax>
    <billingemail>ron@goldenboy.com</billingemail>
    <invoiceline>
      <invoicenum></invoicenum>
      <codetype>Item Code</codetype>
      <linecode>AC-8026</linecode>
      <linedescription>Air Conditioning Unit - Model 26</linedescription>
      <lineunit>Each</lineunit>
      <invoiceqty>1</invoiceqty>
      <invoiceunitprice>1875</invoiceunitprice>
      <customerunitprice>1875</customerunitprice>
      <invoiceunittax>234.375</invoiceunittax>
      <discountpercent>0</discountpercent>
      <discountamount>0</discountamount>
      <extendednetprice>1875</extendednetprice>
      <extendedtax>234.38</extendedtax>
      <extendedtotalprice>2109.38</extendedtotalprice>
      <priceoverride>False</priceoverride>
      <taxcode>GST</taxcode>
    </invoiceline>
  </invoiceheader>
</salesinvoice>
```

Sample URL:

<http://localhost:82/salesinvoice?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

<https://domainname:port/salesinvoice?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&configuration=0>

Results: The Responsevalue contains the Direct Invoice Number(s) created.

```
<?xml version="1.0" encoding="utf-8"?>
<ostendoapi>
  <ostendoapiversion>1.0.0.88</ostendoapiversion>
  <responsestatus>ok</responsestatus>
  <responsevalue>500083</responsevalue>
</ostendoapi>
```

6. Tabledata

- Specifying this resource in your POST call will send the data into an Ostendo table.
- **TableName:** This is a required parameter.
- **Keyfield:** This is a required parameter.
(Note: If you are using SYSUNIQUEID as the keyfield, and you wish to insert new records, the sysuniqueid specified in your content should be zero (<sysuniqueid>0</sysuniqueid>))
- **Content:** Record data in XML format only. Content can include one or more records.

Sample Content:

```
<?xml version="1.0" encoding="utf-8"?>
<ostendoimport>
  <standardunits>
    <standardunit>Pound</standardunit>
    <unitdescription>Pound</unitdescription>
  </standardunits>
  <standardunits>
    <standardunit>Ounce</standardunit>
    <unitdescription>Ounce</unitdescription>
  </standardunits>
</ostendoimport>
```

Sample URL:

<http://localhost:82/tabledata/?tablename=standardunits&keyfield=standardunit&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D>

<https://domainname:port/tabledata/?tablename=standardunits&keyfield=standardunit&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D>

Results: The Responsevalue contains the number of records added or updated.

```
<?xml version="1.0" encoding="utf-8"?>
<ostendoapi>
  <ostendoapiversion>1.0.0.88</ostendoapiversion>
  <responsestatus>ok</responsestatus>
  <responsevalue>RowsAffected: 2</responsevalue>
</ostendoapi>
```

Note: You can get a report listing all the tables in the Ostendo database by going to the **General menu – Reports – Full Listing of Tables.**

This report allows you to select one or more tables to be printed or previewed on screen. The report lists all the fieldnames, field types, length, and any Required fields in each table.

This report is useful if your application needs to query or update any tables in Ostendo.

C. Using the DELETE Method

Using the HTTP or HTTPS DELETE method, you can delete records from an Ostendo table.

- TableName : This is a required parameter.
- Condition=*fieldname=fieldvalue* : This is a required parameter

Keyfield, format, and content are not required.

Sample URL:

<http://localhost:82/tabledata?tablename=TIMESHEETLINES&condition=SYSUNIQUEID=870145&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D>

<https://domainname:port/tabledata?tablename=TIMESHEETLINES&condition=SYSUNIQUEID=870145&apikey=SxQqzOZpWDjQZvEHwAg%3D%3D>

Results: The Rowaffected value contains the number of records deleted.

```
<?xml version="1.0" encoding="utf-8" ?>
<ostendoapi>
<ostendoapiversion>2.0.0.378</ostendoapiversion>
<rowaffected>1</rowaffected>
</ostendoapi>
```

D. Special Resources

Users or applications which are authorised to perform SQL queries via the Ostendo API Service can use the HTTP or HTTPS POST method to submit valid SQL queries to one or more tables in the Ostendo database.

For example, a single SQL query can be used to extract specific data elements or summarised data from one or more tables.

sqlquery

- Specifying this resource in your POST call will run the SQL statement specified in the Content.
- Output Format: xml or json or raw
- Delimiter: comma or tab (for raw output only)
- Content: SQL statement.

Sample SQL statement:

```
Select jobheader.ordernumber, linecode,linedescription,orderqty from jobheader,joblines where jobheader.ordernumber = joblines.ordernumber and jobheader.ordernumber = 'JOB400004'
```

Sample URL:

<http://localhost:82/sqlquery?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=json&configuration=0>

<https://domainname:port/sqlquery?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=json&configuration=0>

Results:

```
[
  {
    "ORDERNUMBER":"JOB400004",
    "LINECODE":"MATERIAL",
    "LINEDESCRIPTION":"Material Used in Progress Claim",
    "ORDERQTY":1
  },
  {
    "ORDERNUMBER":"JOB400004",
    "LINECODE":"LAB-INSPECTION",
    "LINEDESCRIPTION":"Standard QA Labour",
    "ORDERQTY":0
  }
]
```

Sample URL: (format=raw; if delimiter is not specified, then comma is the default delimiter)

<http://localhost:82/sqlquery?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=raw&delimiter=comma>

<https://domainname:port/sqlquery?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=raw&delimiter=comma>

Results:

```
JOB400004,MATERIAL,Material Used in Progress Claim,1
JOB400004,LAB-INSPECTION,Standard QA Labour,0
```

Sample URL: (format=xml; tablename must be specified if format=xml)

<http://localhost:82/sqlquery?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=xml&tablename=jobheader>

<https://domainname:port/sqlquery?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=xml&tablename=jobheader>

Results:

```
<?xml version="1.0" encoding="utf-8"?>
<ostendoexport>
  <jobheader>
    <ordernumber>JOB400004</ordernumber>
    <linecode>MATERIAL</linecode>
    <linedescription>Material Used in Progress Claim</linedescription>
    <orderqty>1</orderqty>
  </jobheader>
```



```
<jobheader>
  <ordernumber>JOB400004</ordernumber>
  <linecode>LAB-INSPECTION</linecode>
  <linedescription>Standard QA Labour</linedescription>
  <orderqty>0</orderqty>
</jobheader>
</ostendoexport>
```

For **UPDATE** or **INSERT** statements, use ***executesql*** instead of sqlquery.

Sample URL:

<http://localhost:82/executesql?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=json&configuration=0>

<https://domainname:port/executesql?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=json&configuration=0>

Sample SQL: ***update*** salesheader set purchasereference = 'abc456' where ordernumber = 'DO300315'

Sample URL: (format=***xml***; ***tablename*** must be specified if format=***xml***)

<http://localhost:82/executesql?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=xml&tablename=standardproperties>

<https://domainname:port/executesql?apikey=SxQqzOZpWDjQZvEHwAg%3D%3D&format=xml&tablename=standardproperties>

Sample SQL: ***insert into*** STANDARDPROPERTIES (PROPERTYNAME,PROPERTYTYPE,PROPERTYVALUES) values('Shape','Text','') '

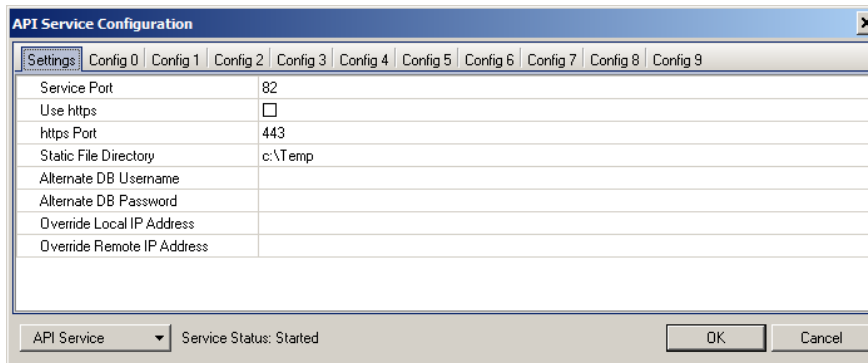
E. Configuring the Ostendo API Service

Note: Always run Ostendo “As Administrator” before trying to configure the API service.

For enhanced security, it is strongly recommended that HTTPS be used in preference to HTTP. This is because all HTTPS sessions are encrypted. Use HTTP only if the Http requests are not routed via the internet.

The Ostendo API Service is easily configured by following the following steps.

1. Go to FILE → API Service → API Configuration



- a. In the **Settings** tab, fill out the following:

- | | | |
|------------------------------|---|---|
| Service Port | - | For HTTP : specify the network port which is dedicated to this API service |
| Use https | - | Tick this box if you are using HTTPS |
| https Port | - | For HTTPS only. Default is 443. |
| Static File Directory | - | Specify the directory for the Static File folder. |

This folder is where you can store html pages which can be accessed by the API user.

- | | | |
|--------------------------------|---|---|
| Alternate DB Username- | - | If the Firebird DB Username is not SYSDBA, then enter it here. |
| Alternate DB Password - | - | If the Firebird DB Password is not the default, then enter it here. |

- | | | |
|------------------------------------|---|--|
| Override Local IP Address - | - | For HTTP : If there are multiple IP addresses for the server, enter the specific IP address to be used when connecting locally. |
|------------------------------------|---|--|

- | | | |
|-------------------------------------|---|---|
| Override Remote IP Address - | - | For HTTP : If there are multiple IP addresses for the server, enter the specific IP address to be used when connecting remotely. |
|-------------------------------------|---|---|

For **HTTPS**, you need the following in your API Service Configuration:

- | | | |
|----------------------------|---|--|
| Use HTTPS | - | tick this box |
| Https Port | - | specify the HTTPS Port number. (Default is 443) |
| Override Local IP Address | - | Enter the Domain Name which is SSL Certified |
| Override Remote IP Address | - | Enter the Domain Name which is SSL Certified |

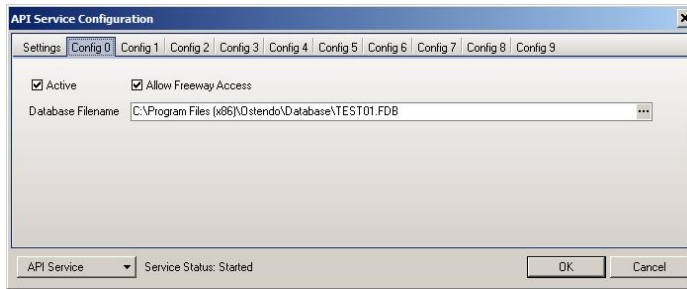
For **HTTPS**, the SSL certificate needs to be installed and configured using Windows **http.sys**

Note: If you want the API to listen to **both Http and Https** ports, then you need to modify the ostendoapi.ini file as follows:

- a. Add the parameter “**UseHttpWithHttps=1**” to the file.
- b. Set the parameter “**OverrideIP=**” to be the **local IP address**

The “**OverrideIPRemote=**” should point to the Domain Name which is SSL certified

b. Next, go to **Config 0** tab:



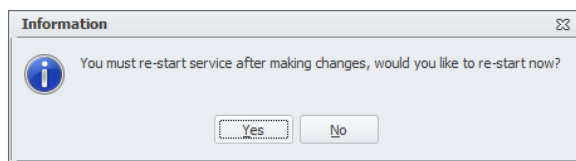
Here you enter the path of the Ostendo Database you wish the API Service to have default access to. Tick the **Active** box to activate this configuration.

You can specify multiple databases (up to 9 additional) using the rest of the Config tabs. This is useful if you want to set up one or more databases for training/development purposes in addition to the LIVE database.

Allow Freeway Access should be ticked if you are also implementing Freeway Mobility solutions.

*Note: Only **Config 0** can be used for Freeway.*

2. Click OK once you have finished making the changes. You will be prompted to Re-Start the API service.



Click "Yes".

3. Firewalls and Port Forwarding

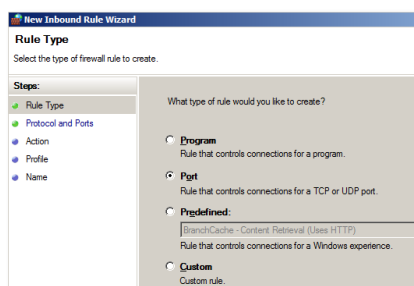
Note: Please ensure your Windows Firewall is set to allow access to the Ostendo API port.

The following example is based on Windows 7. Other versions may be slightly different. The objective is to ensure that the Firewall allows incoming and outgoing traffic via the designated Ostendo API port.

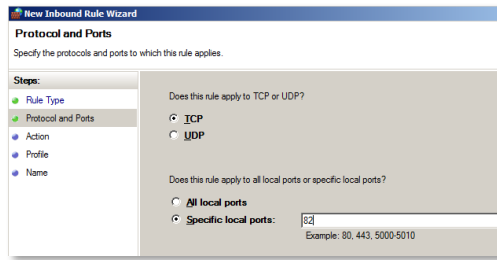
- Go to Windows Firewall – Advanced Settings:



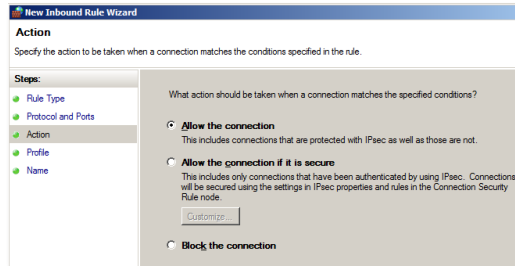
- Click on **Inbound Rules, New Rule...**
- **Rule Type:** Port. Click Next.



- **Protocol and Ports:** TCP, Specific local port = designated Ostendo API port number. Click Next.



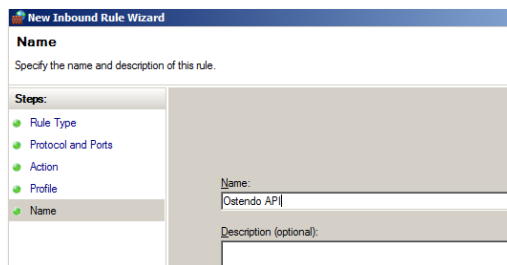
- **Action:** Allow the connection. Click Next.



- **Profile:** Applies to Domain, Private, Public. Click Next again.



- **Name:** Give the Rule an appropriate name. Click Finish.



- Next, click on **Outbound Rules** and do exactly the same as above.

IP Address and Port Forwarding (Applicable for Remote connections only):

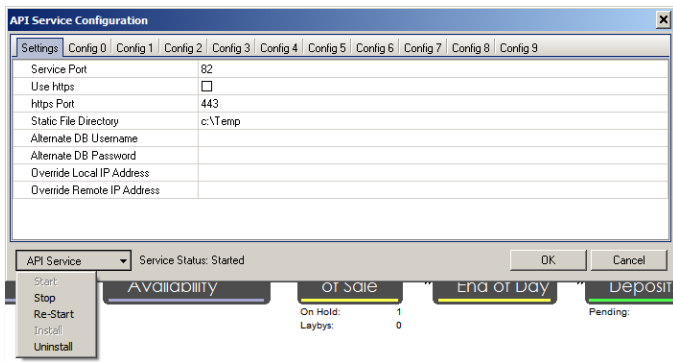
You will need to assign a fixed (internal) **IPv4 address** to your Ostendo server and set up **Port Forwarding** on your router so that all calls to that API port will be forwarded to your Ostendo server correctly.

A Static IP Address (Fixed External IP Address) is required if you have other 3rd party applications interfacing with your Ostendo API.

Ostendo Freeway does NOT require your Ostendo API to have a Static IP Address.

F. START and STOP the API Service

You can start and stop, and re-start the API Service at any time by using the API Service button in the API Configuration screen.

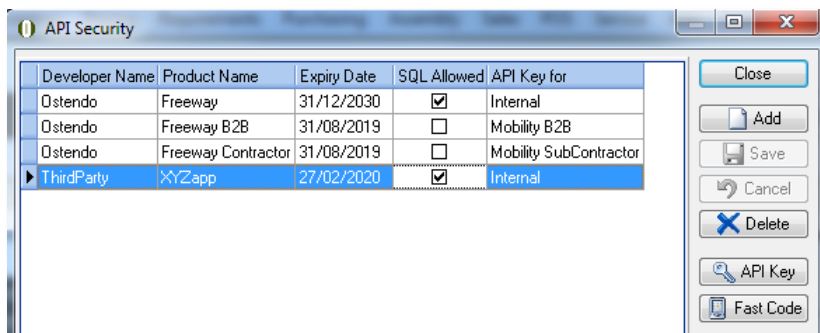


This is where you can **Uninstall** and **Install** the API Service as well.

G. API Security

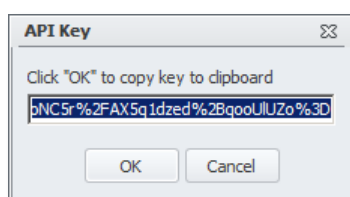
The API Security screen allows you to authorize one or more third-party developers and products to have access to your Ostendo API Service. It also enables you to generate the API key for each developer-product.

1. Go to File → API Service → API Security



- a. Click ADD button to add a new entry.
- b. Fill in Developer Name, Product Name
- c. Select Expiry Date for this authorization
- d. Tick SQL Allowed if this Application is allowed to make SQL queries
- e. Click SAVE button to save the record.
- f. Click **API Key** button to generate the API Key for this Application

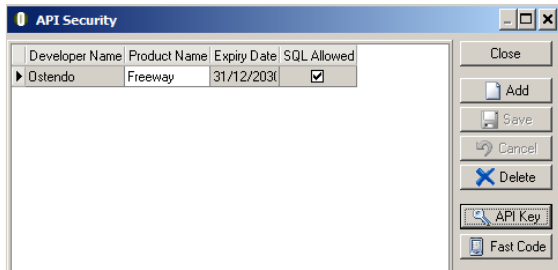
The API Key is URL encoded.



Click OK to copy the generated key and paste it in a notepad or email to be sent to the authorized user.

This key is not stored in Ostendo. However it can be generated again by clicking on the API key button when required.

- g. The **FastCode** button is used for Freeway Mobility only.



This will generate a short code which can be sent to a mobility user. When the mobility user keys in this Fast Code into the app, it will pull the actual API Key and the Freeway License key into the app. This simplifies the process of loading the Keys into the Freeway mobility app.

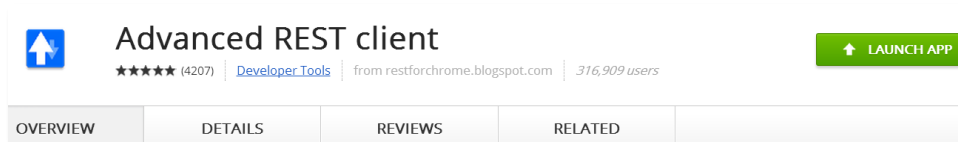
H. Table Authorization **(not implemented yet)**

Apart from access authorization via the API Key, Ostendo API Service provides another level of security whereby users can be restricted to access only specified tables and using approved HTTP methods only.

APPENDIX A – Using Google’s Advanced Rest Client to test your HTTP calls

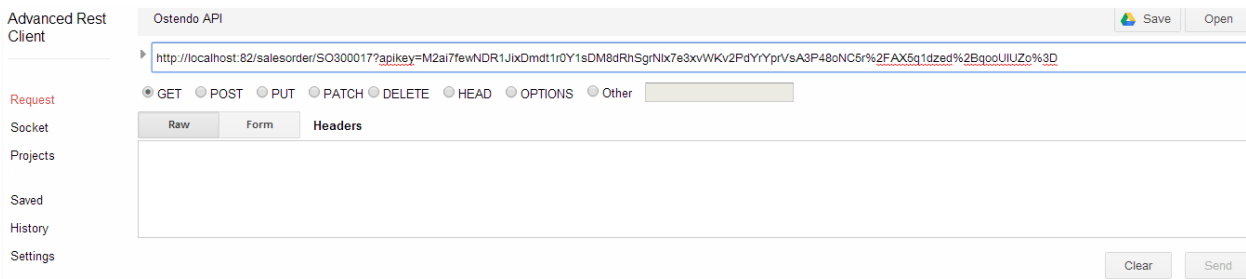
The Advanced Rest Client is a free app/tool you can use to test your API calls to the Ostendo API Service. This tool helps you to ensure that your HTTP calls are properly formed. (POSTMAN is another tool you can use).

When your Ostendo API service is set up and your API key have been generated, you can start using this tool. To begin, do a google search for “Advanced Rest Client”.

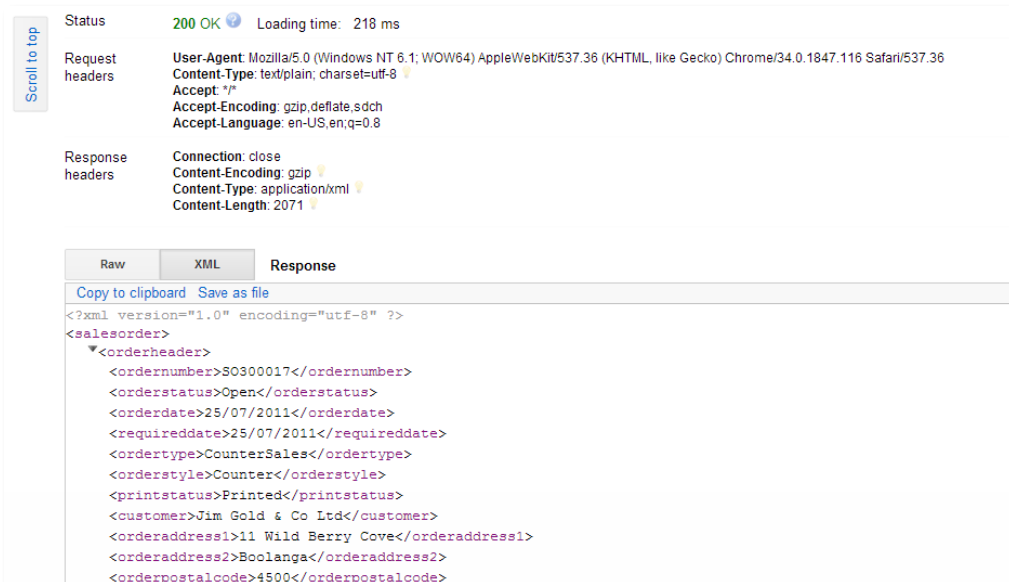


Click on “Launch App” button to launch the application.

- a. To try out a HTTP GET request, you select GET radio button, fill in the URL, and click on SEND button to send your request:



And you should get results like this:



- Status 200 OK means the call was successful.
- The XML at the bottom is what will be returned to the calling application.

- a. To try out a HTTP POST request, you select POST radio button, fill in the URL, the Content in the Payload section, and click on SEND button to send your request:
Please note that the Content-Type should be set to “application/xml”

The screenshot shows an HTTP client interface with the following details:

- URL: `http://localhost:82/tabledata?tablename=standardunits&keyfield=standardunit&apikey=M2ai7fewNDR1JixDmdt1r0Y1sDM8dRhSgrNlx7e3xvWkV2PdYyYprVsA3P48oNC5r%2FAX5q1dzed%2BqooUIUz08`
- Method: POST (selected)
- Headers: Raw, Form, Headers
- Payload: Raw, Form, Files (0), Payload
- Content-Type: application/xml (selected)
- Buttons: Clear, Send

```
<?xml version="1.0" encoding="utf-8"?>
<ostendoimport>
<standardunits>
  <standardunit>Pound</standardunit>
  <unitdescription>Pound</unitdescription>
</standardunits>
<standardunits>
  <standardunit>Ounce</standardunit>
  <unitdescription>Ounce</unitdescription>
</standardunits>
</ostendoimport>
```

And you should get results like this:

The screenshot shows the response details and raw response in an HTTP client:

- Status: 200 OK (with a blue checkmark icon) Loading time: 91 ms
- Request headers: User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/34.0.1847.116 Safari/537.36; Origin: chrome-extension://hgml0ofddfdnphgcellkdfbfjeloo; Content-Type: application/xml; Accept: */*; Accept-Encoding: gzip, deflate, sdch; Accept-Language: en-US,en;q=0.8
- Response headers: Connection: close; Content-Encoding: gzip; Content-Type: text/html; charset=ISO-8859-1; Content-Length: 140
- Response: Raw, Parsed, Response

```
<?xml version="1.0" encoding="utf-8"?>
<ostendoapi>
<ostendoapiversion>1.0.0.88</ostendoapiversion>
<responsestatus>ok</responsestatus>
<responsevalue>RowsAffected: 2</responsevalue>
</ostendoapi>
```

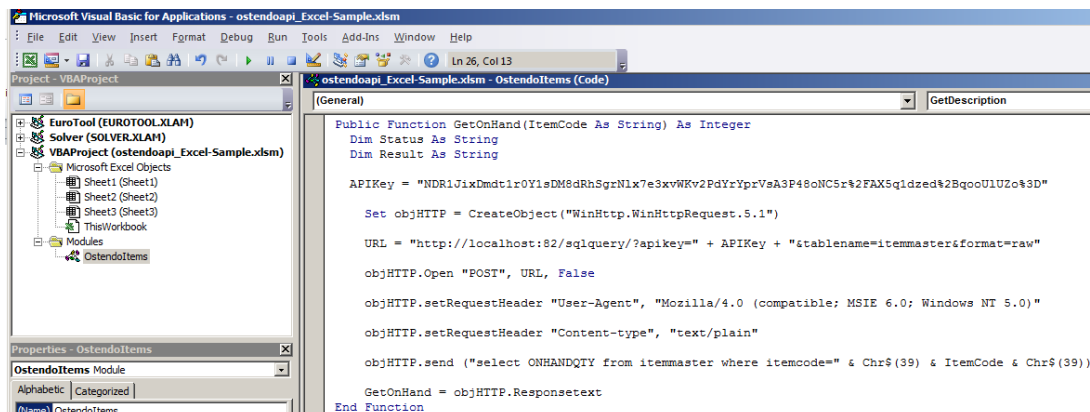
- Status 200 OK means the request was successful.
- The responsevalue “RowsAffected: 2” means two records were added to the table.

Advanced Rest Client allows you to save these tests as a Project (using the SAVE button) and you can give it a name like “Ostendo API”.

If you save it, then the next time you launch Advanced Rest Client, your last request will be displayed and you can carry on testing.

APPENDIX B – A Simple example of Excel spreadsheet getting data via the Ostendo API Service

1. Copy the following code into a new macro in the Excel Workbook. Replace the API key and Port No. with the appropriate data.



```
Public Function GetOnHand(ItemCode As String) As Integer
    Dim Status As String
    Dim Result As String

    APIKey = "NDRLJixDmdct1r0Y1sDM8dRhSqrNlx7e3xvWKv2PdYrYprVsA3P48oNC5r%2FAX5qldzed%2BqooU1UZc%3D"

    Set objHTTP = CreateObject("WinHttp.WinHttpRequest.5.1")

    URL = "http://localhost:82/sqlquery/?apikey=" + APIKey + "&tablename=itemmaster&format=raw"

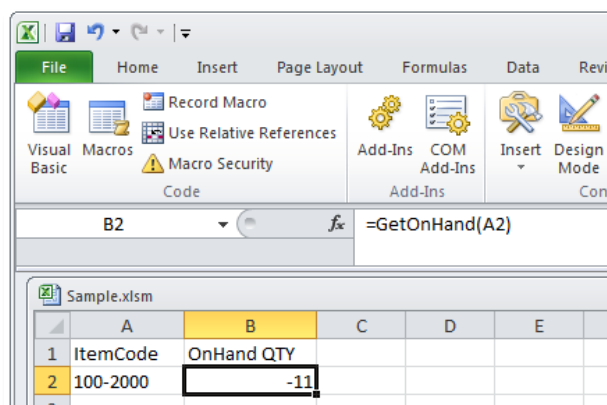
    objHTTP.Open "POST", URL, False

    objHTTP.setRequestHeader "User-Agent", "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)"
    objHTTP.setRequestHeader "Content-type", "text/plain"

    objHTTP.send ("select ONHANDQTY from itemmaster where itemcode=" & Chr$(39) & ItemCode & Chr$(39))

    GetOnHand = objHTTP.ResponseText
End Function
```

- a. Please note that the API Key and Port number must be correct.



Column A is ItemCode; column B is OnHandQty

If an item code is keyed into Cell (A,2); then cell B2 (=GetOnHand(A2)) will display the Onhand Qty .

