

SF-LWR Extension Framework : Vertex Tax Integration

[Functional Overview](#)

[Technical Details](#)

[Sequence Diagram](#)

[Configuration details](#)

[Record - Vertex \(Tax Provider \)](#)

[Record - Vertex Tax Calc Service \(HTTP Service\)](#)

[Source Code](#)

Functional Overview

This document provides details specific to integrating with Vertex.

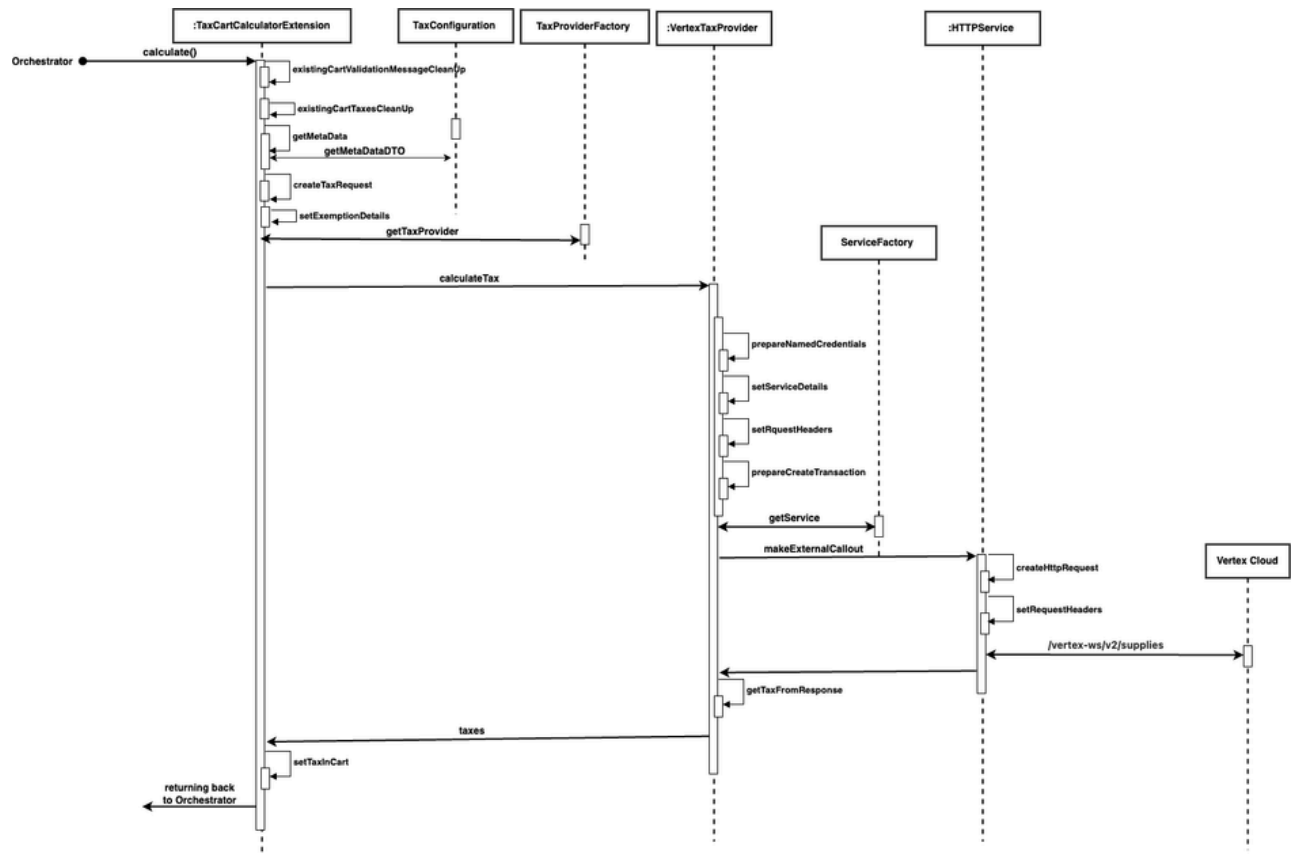
For common details you can refer to parent doc [here](#).

Technical Details

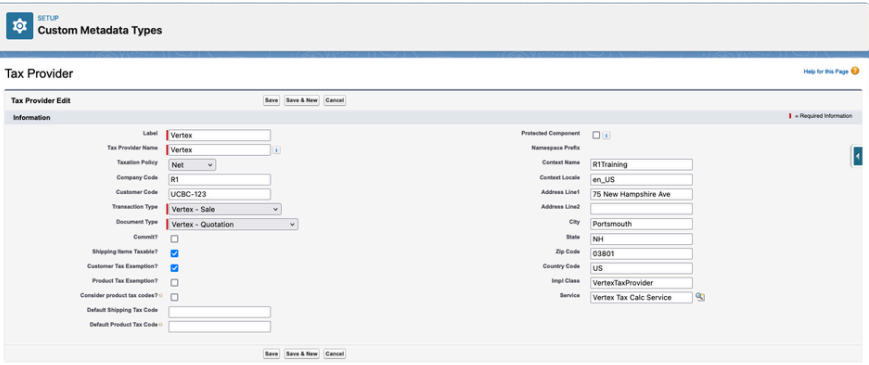
This integration implementation has two parts :

- **Configuration**
 - Custom Meta Data Records
 - Vertex
 - Vertex Tax Calc Service
 - Named Credentials
 - Vertex Credentials
- **Source Code**
 - VertexTaxProvider
 - VertexTaxRequest
 - VertexTaxResponse

Sequence Diagram



Configuration details

	Section	Task	Details
1	Salesforce Org	Tax Provider - Create a record for Vertex	<p>Create a record Vertex for Custom Meta Data Type "Tax Provider"</p> 
2	Salesforce Org	HTTP Service - Create a record for Vertex	<p>Create a record - Vertex Tax Calc Service of Custom Meta Data Type "HTTP Service"</p>

SETUP Custom Metadata Types

HTTP Service

HTTP Service Edit

Information

Label: Vertex Tax Calc Service

HTTP Service Name: Vertex_Tax_Calc_Service

Protected Component: ☐

Impl Class:

Named Credentials: Vertex_Credentials

Service Method: Live

HTTP Method: POST

End Point: /vertex-smb/v2/supplies

Service Timeout: 5000

Success Codes: 200,301

Error Codes: 400,403,404,500,503

Auth Codes: 401

Mocked Response:

Namespace Prefix:

3 Salesforce Org

Create External Credentials

Create **External Credentials** for OAuth 2.0,

SETUP Named Credentials

Named Credentials External Credentials

3 Items - Sorted by Label

Label	Authentication Protocol	Actions
FedEx Authentication	OAuth 2.0	New
UPS Authentication	OAuth 2.0	New
Vertex External Credentials	OAuth 2.0	New

Provide details , like

Name - Vertex_External_Credentials

Authentication Protocol - AOAuth 2.0

Authentication Flow Type - Client Credentials with Secret Flow

Identity Provider URL - <https://auth.vertexsmb.com/identity/connect/token>

Edit Vertex External Credentials

*Label: Vertex External Credentials

*Name: Vertex_External_Credentials

*Authentication Protocol: OAuth 2.0

*Authentication Flow Type: Client Credentials with Client Secret Flow

Scope:

*Identity Provider URL: <https://auth.vertexsmb.com/identity/connect/token>

☒ Pass client credentials in request body

Cancel Save

SETUP > NAMED CREDENTIALS

Vertex External Credentials

EditDelete

Label

Vertex External Credentials

Name

Vertex_External_Credentials

Authentication Protocol

OAuth 2.0

Authentication Flow Type

Client Credentials with Client Secret Flow

Identity Provider URL

https://auth.vertexmb.com/identity/connect/token

Scope

Managed Package Access

Created By Namespace

Related Named Credentials

Label

Name

URL

Vertex Credentials

Vertex_Credentials

https://calconnect.vertexsmb.com

Principals

Sequ...

Parameter Name

Client ID

Authentication Status

Actions

1

VertexPrincipal

fa2e397fe8604546bd5602b1143e4863

Configured

Also create a Principal which will store Client ID & Client Secret

Name - VertexPrincipal

Edit Principal

* Parameter Name

VertexPrincipal

* Sequence Number

1

Client ID

fa2e397fe8604546bd5602b1143e4863

Client Secret

.....

Principal Access

Name

B2BBuyer

Entity Type

PermissionSet

ID

OPSHn000001yYV7OAM

Cancel

Save

4

Salesforc
e Org

Create Named
Credentials

Create a Named Credentials for storing Vertex API details

Set URL - <https://calconnect.vertexsmb.com/>

Turn on - Generate Authorization Header

SETUP > NAMED CREDENTIALS

Vertex Credentials

EditDelete

Label

Vertex Credentials

Name

Vertex_Credentials

URL

https://calconnect.vertexsmb.com

Enabled for Callouts

Authentication

External Credential

Vertex External Credentials

Client Certificate

Callout Options

Generate Authorization Header

Allow Formulas in HTTP Header

Allow Formulas in HTTP Body

Outbound Network Connection

Managed Package Access

Created By Namespace

Allowed Namespaces for Callouts

5

Salesforce Org

Assign Permission

To make callouts that use a named credential, you must enable principal access to permission sets or profiles for the users who need access.

Setup > Permission Sets > B2BBuyer

Click **External Credential Principal Access**

SETUP

Permission Sets

Permission Set

B2BBuyer

Find Settings...CloneEdit PropertiesManage AssignmentsView Summary (Beta)

Permission Set Overview

Description

Allows access to the store. Lets users see products and categories, make purchases, and add products to a wishlist.

API Name

B2BBuyer

Licenses

1

B2B Buyer Permission Set One Seat

Namespace Prefix

Session Activation Required

☐

Created By

Mass Tsgg, 2/9/2024, 11:25 AM

Permission Set Groups Added To

0

Last Modified By

Mass Tsgg, 4/2/2024, 9:05 PM

Apps

Object Settings

Permissions to access objects and fields, and settings such as tab availability

App Permissions

Permissions to perform app-specific actions, such as "Manage Call Centers"

Flow Access

Permissions to execute Flows

External Credential Principal Access

Permissions to authenticate with external credential principal mappings

Custom Metadata Types

Permissions to access custom metadata types

Custom Setting Definitions

Permissions to access custom settings

System

Settings that apply across all apps, such as record and user management

Learn More

System Permissions

Permissions to perform actions that apply across apps, such as "Modify All Data"

SETUP

Permission Sets

Permission Set

B2BBuyer

Find Settings...CloneEdit PropertiesManage AssignmentsView Summary (Beta)

Permission Set Overview

External Credential Principal Access

External Credential Principal Access

Save

Close

Available External Credential Principals

Vertex_External_Credentials - VertexPrincipal

Enabled External Credential Principals

FedExAuth - FedExCredentials
UPSAuth - UPSCredentials

Add

Remove

SETUP

Permission Sets

Permission Set

B2BBuyer

Find Settings...CloneEdit PropertiesManage AssignmentsView Summary (Beta)

Permission Set Overview

External Credential Principal Access

Items per page: 50 Items

View: All

External Credential Principal Access

Edit

External Credential Principal Name

Installed Package

FedExAuth - FedExCredentials

UPSAuth - UPSCredentials

Vertex_External_Credentials - VertexPrincipal

More details can be found here - https://help.salesforce.com/s/articleView?id=sf.nc_enable_ext_cred_principal.htm&type=5

Record - Vertex (Tax Provider)

Field Label	Field Value
Context Name	<Webstore name> e.g. R1Training
Context Locale	<locale> e.g. en_US
Impl Class	VertexTaxProvider
Address Line1	75 New Hampshire Ave
Address Line2	
City	Portsmouth
State	NH
Zip Code	03801
Country Code	US
Company Code	R1
Customer Code	UCBC-123
Taxation Policy	Net Gross
Transaction Type	Vertex - Sale Vertex - Rental Vertex - Lease
Document Type	Vertex - Quotation Vertex - Invoice Vertex - Distribute
Commit?	True/False
Shipping Items Taxable?	True/False
Consider product tax codes?	True/False
Customer Tax Exemption?	True/False
Product Tax Exemption?	True/False
Default Product Tax Code	
Default Shipping Tax Code	
Service	Vertex Tax Calc Service

Record - Vertex Tax Calc Service (HTTP Service)

Field Label	Field Value
-------------	-------------

End Point	/vertex-ws/v2/supplies
HTTP Method	POST
Impl Class	<i>This is optional so could be left blank</i>
Named Credentials	Vertex_Credentials
Service Mode	Live Mocked
Timeout	5000 (ms)
Success Codes	200,201
Error Codes	400,403,404,500,503
Auth Codes	401
Mocked Response	N/A

Source Code

Class	Details	Code
VertexTaxProvider	<p>This class extends ShippingProvider class & implements following methods -</p> <p>prepareCreateTransaction</p> <p>prepareCommitTransaction</p> <p>getTaxFromResponse</p>	<pre> 1 public without sharing class VertexTaxProvider extends TaxProv 2 public VertexTaxProvider() { 3 } 4 5 public override void prepareCreateTransaction(6 TaxProviderRequest taxRequest, 7 Map<String, String> callOutRequest 8) { 9 Boolean hasMultipleShipments = taxRequest.hasMultipleShipm 10 taxRequest.hasMultipleShipments == true 11 ? true 12 : false; 13 14 15 VertexTaxRequest vertexRequest = new VertexTaxRequest(); 16 vertexRequest.transactionType = taxRequest.taxTransacionTy 17 vertexRequest.saleMessageType = taxRequest.taxDocType; 18 19 //setting currency 20 VertexTaxRequest.cls_currency currencyZ = new VertexTaxReq 21 currencyZ.isoCurrencyCodeAlpha = taxRequest.currencyCode; 22 vertexRequest.currency_z = currencyZ; 23 24 // setting current date 25 DateTime todaysDate = System.today(); 26 String todaysDateStr = todaysDate.format('yyyy-MM-dd'); 27 vertexRequest.documentDate = todaysDateStr; 28 vertexRequest.taxPointDate = todaysDateStr; 29 30 vertexRequest.documentNumber = String.isNotBlank(taxRequest 31 vertexRequest.transactionId = String.isNotBlank(taxRequest. 32 if (33 taxRequest.taxableCartItems != null && 34 taxRequest.taxableCartItems.keySet().size() > 0 </pre>

```

35     ) {
36         vertexRequest.lineItems = prepareRequestFromExistingData(
37             taxRequest,
38             hasMultipleShipments
39         );
40     } else {
41         vertexRequest.lineItems = prepareRequestFromDB(
42             taxRequest,
43             hasMultipleShipments
44         );
45     }
46
47     //in case there is single shipment , setting addresses at
48     if (!hasMultipleShipments) {
49         vertexRequest.customer = setCustomerDetails(
50             taxRequest,
51             taxRequest.street,
52             taxRequest.city,
53             taxRequest.state,
54             taxRequest.postalCode,
55             taxRequest.country
56         );
57         vertexRequest.seller = setSellerDetails(taxRequest);
58     }
59
60     String vertexRequestBody = JSON.serialize(vertexRequest, t
61     vertexRequestBody = vertexRequestBody.replaceAll('_', '')
62     callOutRequest.put(Constants.SERVICE_REQUEST_BODY, vertexR
63     System.debug('===== vertexRequest : '+vertexRequestBody);
64 }
65
66 public virtual override TaxProviderResponse getTaxFromRespon
67     String strvertexResponseBody,
68     TaxProviderRequest tpRequest
69 ) {
70     TaxProviderResponse tpResponse = new TaxProviderResponse()
71     Map<String, LineItemTaxDetails> taxes = new Map<String, Li
72     Map<String, TaxableCartItem> taxableCartItems = tpRequest.
73
74     TaxProviderResponse shippingResponse;
75     System.debug('===== strvertexResponseBody : '+strvertexRe
76     if (String.isNotBlank(strvertexResponseBody)) {
77         VertexTaxResponse vertexTaxResponse = VertexTaxResponse.
78             strVertexResponseBody
79     );
80     if (
81         vertexTaxResponse.data.lineItems != null &
82         vertexTaxResponse.data.lineItems.size() > 0
83     ) {
84         for (VertexTaxResponse.cls_lineItems line : vertexTaxR
85             String lineItemId = line.lineItemId;
86             LineItemTaxDetails liTaxDetails = new LineItemTaxDet
87             liTaxDetails.tax = 0;
88             liTaxDetails.rate = 0;
89             if (line.totalTax > 0) {
90                 liTaxDetails.tax = line.totalTax;
91             }
92             if (line.fairMarketValue > 0) {
93                 liTaxDetails.rate = liTaxDetails.tax / line.fairMa
94             }

```



```

95         taxableCartItems.get(lineItemId).lineItemTexas.add(l
96     }
97     tpResponse.taxableCartItems = taxableCartItems;
98 }
99 }
100 return tpResponse;
101 }
102
103 private List<VertexTaxRequest.cls_lineItems> prepareRequestF
104     TaxProviderRequest taxRequest,
105     Boolean hasMultipleShipments
106 ) {
107     List<VertexTaxRequest.cls_lineItems> lines = new List<Vert
108     for (String id : taxRequest.taxableCartItems.keySet()) {
109         TaxableCartItem cartItem = taxRequest.taxableCartItems.g
110         VertexTaxRequest.cls_lineItems line = prepareLine(
111             id,
112             cartItem.sku,
113             cartItem.amount,
114             cartItem.lineItemType,
115             cartItem.quantity,
116             cartItem.taxClassId,
117             cartItem.productExemptionCode,
118             cartItem.entityUseCode,
119             taxRequest,
120             false
121         );
122
123         if (hasMultipleShipments) {
124             line.seller = setSellerDetails(taxRequest);
125             line.customer = setCustomerDetails(
126                 taxRequest,
127                 cartItem.street,
128                 cartItem.city,
129                 cartItem.state,
130                 cartItem.postalCode,
131                 cartItem.country
132             );
133         }
134         lines.add(line);
135     }
136     return lines;
137 }
138
139 private List<VertexTaxRequest.cls_lineItems> prepareRequestF
140     TaxProviderRequest taxRequest,
141     Boolean hasMultipleShipments
142 ) {
143
144     String cartId = taxRequest.cartId;
145     taxRequest.taxableCartItems = new Map<String, TaxableCartI
146     List<VertexTaxRequest.cls_lineItems> lines = new List<Vert
147     String query = 'SELECT Id, Sku, TotalLineAmount, Quantity,
148     if (hasMultipleShipments) {
149         query += ', CartDeliveryGroup.Id, CartDeliveryGroup.Delive
150     }
151     if(taxRequest.taxMetaData.useProductTaxCodes){
152         query += ', Product2.Tax_Class_Id__c';
153     }
154     if(taxRequest.taxMetaData.productTaxExemption){

```

```

155     query += ',Product2.Taxable__c,Product2.Entity_Use_Code_
156 }
157 query += ' FROM CartItem WHERE  cartId=:cartId';
158 if (!taxRequest.taxMetaData.shippingItemsTaxable) {
159     query += ' AND Type = \'Product\'';
160 }
161 for (CartItem cartItem : Database.query(query)) {
162     VertexTaxRequest.cls_lineItems line = prepareLine(
163         cartItem.Id,
164         cartItem.SKU !=null ? cartItem.SKU : cartItem.Product2
165         cartItem.TotalLineAmount,
166         cartItem.Type,
167         cartItem.Quantity,
168         taxRequest.taxMetaData.useProductTaxCodes && cartItem.
169         taxRequest.taxMetaData.productTaxExemption ? cartItem.
170         taxRequest.taxMetaData.productTaxExemption ? cartItem.
171         taxRequest,
172         true
173     );
174     if (hasMultipleShipments) {
175         line.seller = setSellerDetails(taxRequest);
176         line.customer = setCustomerDetails(
177             taxRequest,
178             cartItem.CartDeliveryGroup.DeliverToStreet,
179             cartItem.CartDeliveryGroup.DeliverToCity,
180             cartItem.CartDeliveryGroup.DeliverToState,
181             cartItem.CartDeliveryGroup.DeliverToPostalCode,
182             cartItem.CartDeliveryGroup.DeliverToCountry
183         );
184     }
185     lines.add(line);
186 }
187 return lines;
188 }
189
190 private VertexTaxRequest.cls_lineItems prepareLine(
191     ID id,
192     String sku,
193     Decimal taxableAmount,
194     String type,
195     Decimal quantity,
196     String taxClassId,
197     String exemptionCode,
198     String entityUseCode,
199     TaxProviderRequest taxRequest,
200     Boolean setInRequestToo
201 ) {
202     VertexTaxRequest.cls_lineItems line = new VertexTaxRequest
203     line.lineItemId = id;
204     line.extendedPrice = taxableAmount;
205     line.fairMarketValue = taxableAmount;
206     line.usage = type;
207     VertexTaxRequest.cls_product product = new VertexTaxReques
208     product.value = sku;
209     line.product = product;
210     VertexTaxRequest.cls_quantity prdQuantity = new VertexTaxR
211     prdQuantity.value = quantity;
212     line.quantity = prdQuantity;
213     if (setInRequestToo) {
214         TaxableCartItem tcItem = new TaxableCartItem();

```

```

215         tcItem.id = id;
216         tcItem.amount = taxableAmount;
217         tcItem.sku = sku;
218         tcItem.lineItemType = type;
219         taxRequest.taxableCartItems.put(id, tcItem);
220     }
221
222     return line;
223 }
224
225 public override void prepareCommitTransaction(TaxProviderReq
226
227 ) {
228     VertexTaxRequest.cls_seller setSellerDetails(
229         TaxProviderRequest taxRequest
230     ) {
231         VertexTaxRequest.cls_seller seller = new VertexTaxRequest.
232         VertexTaxRequest.cls_physicalOrigin origin = new VertexTax
233         origin.streetAddress1 = taxRequest.taxMetaData.shipFromLin
234         origin.city = taxRequest.taxMetaData.shipFromCity;
235         origin.mainDivision = taxRequest.taxMetaData.shipFromState
236         origin.postalCode = taxRequest.taxMetaData.shipFromZipCode
237         origin.country = taxRequest.taxMetaData.shipFromCountry;
238         if (String.isNotBlank(taxRequest.taxMetaData.companyCode))
239             seller.company = taxRequest.taxMetaData.companyCode;
240     }
241     seller.physicalOrigin = origin;
242     return seller;
243 }
244
245 VertexTaxRequest.cls_customer setCustomerDetails(
246     TaxProviderRequest taxRequest,
247     String street,
248     String city,
249     String state,
250     String postalCode,
251     String country
252 ) {
253     VertexTaxRequest.cls_customer customer = new VertexTaxRequ
254     VertexTaxRequest.cls_customerCode customerCd = new VertexT
255     if (String.isNotBlank( taxRequest.taxMetaData.customerCod
256         customerCd.classCode = taxRequest.taxMetaData.customer
257         customerCd.isBusinessIndicator = true;
258         customerCd.value = taxRequest.taxMetaData.customerCode;
259         customer.customerCode = customerCd;
260     }
261
262     if (taxRequest.taxMetaData.customerTaxExemption){
263         if (String.isNotBlank(taxRequest.customerExemptionCode))
264             VertexTaxRequest.cls_exemptionCertificate exemptionCer
265             exemptionCertificate.exemptionCertificateNumber = taxR
266             exemptionCertificate.value = taxRequest.customerExempt
267             customer.exemptionCertificate = exemptionCertificate;
268
269         }else{
270             customer.isTaxExempt = true;
271         }
272         if (String.isNotBlank(taxRequest.entityUseCode)) {
273             customer.exemptionReasonCode = taxRequest.entityUseCod
274         }

```

		<pre> 275 } 276 VertexTaxRequest.cls_destination destination = new Ver 277 destination.streetAddress1 = street; 278 destination.city = city; 279 destination.mainDivision = state; 280 destination.postalCode = postalCode; 281 destination.country = country; 282 283 284 customer.destination = destination; 285 return customer; 286 } 287 } </pre>
VertexTaxRequest	This class is a DTO used to prepare Vertex tax calculation request which is posted to UPS to get shipping details	<pre> 1 public without sharing class VertexTaxRequest{ 2 public String accumulationCustomerNumber; 3 public String accumulationDocumentNumber; 4 public String billingType; 5 public cls_companyCodeCurrency companyCodeCurrency; 6 public cls_currency currency_z; 7 public cls_currencyConversionFactors[] currencyConversionF 8 public cls_customer customer; 9 public String deliveryTerm; 10 public cls_discount discount; 11 public String documentDate; 12 public String documentNumber; 13 public String documentSequenceId; 14 public String documentType; 15 public cls_exemptOverrides[] exemptOverrides; 16 public cls_impositionInclusions[] impositionInclusions; 17 public boolean isTaxOnlyAdjustmentIndicator; 18 public cls_lineItems[] lineItems; 19 public String locationCode; 20 public cls_nonTaxableOverrides[] nonTaxableOverrides; 21 public String orderType; 22 public cls_originalCurrency originalCurrency; 23 public String paymentDate; 24 public boolean postToJournal; 25 public String postingDate; 26 public Integer proratePercentage; 27 public cls_rateOverrides[] rateOverrides; 28 public boolean returnAssistedParametersIndicator; 29 public boolean returnGeneratedLineItemsIndicator; 30 public boolean returnTimeElapsedDetailsIndicator; 31 public boolean roundAtLineLevel; 32 public String saleMessageType; 33 public cls_seller seller; 34 public String simplificationCode; 35 public cls_situsOverride situsOverride; 36 public cls_taxOverride taxOverride; 37 public String taxPointDate; 38 public String transactionId; 39 public String transactionType; 40 public class cls_companyCodeCurrency { 41 public String isoCurrencyCodeAlpha; 42 public Integer isoCurrencyCodeNum; 43 public String isoCurrencyName; 44 } 45 public class cls_currency { 46 public String isoCurrencyCodeAlpha; </pre>

```

47         public Integer isoCurrencyCodeNum;
48         public String isoCurrencyName;
49     }
50     public class cls_currencyConversionFactors {
51         public Integer conversionFactor;
52         public cls_sourceCurrency sourceCurrency;
53         public cls_targetCurrency targetCurrency;
54     }
55     public class cls_sourceCurrency {
56         public String isoCurrencyCodeAlpha;
57         public Integer isoCurrencyCodeNum;
58         public String isoCurrencyName;
59     }
60     public class cls_targetCurrency {
61         public String isoCurrencyCodeAlpha;
62         public Integer isoCurrencyCodeNum;
63         public String isoCurrencyName;
64     }
65     public class cls_customer {
66         public cls_administrativeDestination administrativeDes
67         public cls_customerCode customerCode;
68         public cls_destination destination;
69         public cls_exemptionCertificate exemptionCertificate;
70         public String exemptionReasonCode;
71         public boolean isTaxExempt;
72         public cls_taxRegistrations[] taxRegistrations;
73     }
74     public class cls_administrativeDestination {
75         public String city;
76         public String country;
77         public cls_currencyConversion currencyConversion;
78         public String externalJurisdictionCode;
79         public String latitude;
80         public String locationCode;
81         public String locationCustomsStatus;
82         public String longitude;
83         public String mainDivision;
84         public String postalCode;
85         public String streetAddress1;
86         public String streetAddress2;
87         public String subDivision;
88         public String taxAreaId;
89     }
90     public class cls_currencyConversion {
91         public cls_currency currency_z;
92         public Double rate;
93     }
94     public class cls_customerCode {
95         public String classCode;
96         public boolean isBusinessIndicator;
97         public String value;
98     }
99     public class cls_destination {
100         public String city;
101         public String country;
102         public cls_currencyConversion currencyConversion;
103         public String externalJurisdictionCode;
104         public String latitude;
105         public String locationCode;
106         public String locationCustomsStatus;

```

```
107         public String longitude;
108         public String mainDivision;
109         public String postalCode;
110         public String streetAddress1;
111         public String streetAddress2;
112         public String subDivision;
113         public String taxAreaId;
114     }
115     public class cls_exemptionCertificate {
116         public String exemptionCertificateNumber;
117         public String value;
118     }
119     public class cls_taxRegistrations {
120         public boolean hasPhysicalPresenceIndicator;
121         public cls_impositionType impositionType;
122         public String isoCountryCode;
123         public String jurisdictionId;
124         public String mainDivision;
125         public cls_nexusOverrides[] nexusOverrides;
126         public cls_physicalLocations[] physicalLocations;
127         public String taxRegistrationNumber;
128         public cls_filingCurrency filingCurrency;
129         public String taxRegistrationType;
130     }
131     public class cls_impositionType {
132         public String impositionTypeId;
133         public boolean userDefined;
134         public String value;
135         public String withholdingType;
136     }
137     public class cls_nexusOverrides {
138         public boolean city;
139         public boolean country;
140         public boolean district;
141         public String locationRole;
142         public boolean mainDivision;
143         public boolean subDivision;
144     }
145     public class cls_physicalLocations {
146         public String city;
147         public String country;
148         public String mainDivision;
149         public String postalCode;
150         public String streetAddress1;
151         public String streetAddress2;
152         public String subDivision;
153         public String taxAreaId;
154     }
155     public class cls_filingCurrency {
156         public String isoCurrencyCodeAlpha;
157         public Integer isoCurrencyCodeNum;
158         public String isoCurrencyName;
159     }
160     public class cls_discount {
161         public String discountType;
162         public Integer discountValue;
163         public String userDefinedDiscountCode;
164     }
165     public class cls_exemptOverrides {
166         public Integer amount;
```

```

167         public cls_impositionType impositionType;
168         public String jurisdictionType;
169         public String reasonCode;
170     }
171     public class cls_impositionInclusions {
172         public cls_impositionType impositionType;
173         public String jurisdictionType;
174     }
175     public class cls_lineItems {
176         public Double amountBilledToDate;
177         public String chainTransactionPhase;
178         public cls_commodityCode commodityCode;
179         public Double companyCodeCurrencyTaxAmount;
180         public Double companyCodeCurrencyTaxableAmount;
181         public Double cost;
182         public String costCenter;
183         public String countryOfOriginISOCODE;
184         public cls_customer customer;
185         public String deliveryTerm;
186         public String departmentCode;
187         public cls_discount discount;
188         public cls_exemptOverrides[] exemptOverrides;
189         public String exportProcedure;
190         public Double extendedPrice;
191         public Double fairMarketValue;
192         public cls_flexibleFields flexibleFields;
193         public Double freight;
194         public String generalLedgerAccount;
195         public cls_impositionInclusions[] impositionInclusions;
196         public Double inputTotalTax;
197         public String intrastatCommodityCode;
198         public boolean isMulticomponent;
199         public Double landedCost;
200         public String lineItemId;
201         public Integer lineItemNumber;
202         public cls_lineType lineType;
203         public cls_lineTypes[] lineTypes;
204         public String locationCode;
205         public String materialCode;
206         public String materialOrigin;
207         public Integer modeOfTransport;
208         public Integer natureOfTransaction;
209         public Integer netMassKilograms;
210         public cls_nonTaxableOverrides[] nonTaxableOverrides;
211         public cls_product product;
212         public String projectNumber;
213         public cls_quantity quantity;
214         public cls_rateOverrides[] rateOverrides;
215         public cls_returnsFields returnsFields;
216         public cls_seller seller;
217         public String simplificationCode;
218         public cls_situsOverride situsOverride;
219         public cls_statisticalValue statisticalValue;
220         public cls_supplementaryUnit supplementaryUnit;
221         public Double previousTaxPaid;
222         public String taxDate;
223         public boolean taxIncludedIndicator;
224         public cls_taxOverride taxOverride;
225         public String titleTransfer;
226         public String transactionType;

```

```
227         public Double unitPrice;
228         public String usage;
229         public String usageClass;
230         public String vendorSKU;
231         public cls_volume volume;
232         public cls_weight weight;
233     }
234     public class cls_commodityCode {
235         public String commodityCodeType;
236         public String value;
237     }
238     public class cls_flexibleFields {
239         public cls_flexibleCodeFields[] flexibleCodeFields;
240         public cls_flexibleDateFields[] flexibleDateFields;
241         public cls_flexibleNumericFields[] flexibleNumericFields;
242     }
243     public class cls_flexibleCodeFields {
244         public Integer fieldId;
245         public String value;
246     }
247     public class cls_flexibleDateFields {
248         public Integer fieldId;
249         public String value;
250     }
251     public class cls_flexibleNumericFields {
252         public Integer fieldId;
253         public Integer value;
254     }
255     public class cls_lineType {
256         public String accumulationLocation;
257         public String content;
258         public String direction;
259         public String status;
260         public String value;
261     }
262     public class cls_lineTypes {
263         public String accumulationLocation;
264         public String basis;
265         public String content;
266         public Double count;
267         public String direction;
268         public Integer lineTypeNumber;
269         public String status;
270         public String value;
271     }
272     public class cls_nonTaxableOverrides {
273         public Double amount;
274         public cls_impositionType impositionType;
275         public String jurisdictionType;
276         public String reasonCode;
277     }
278     public class cls_product {
279         public String productClass;
280         public String value;
281     }
282     public class cls_quantity {
283         public String unitOfMeasure;
284         public Double value;
285     }
286     public class cls_rateOverrides {
```



```

287         public cls_impositionType impositionType;
288         public String jurisdictionType;
289         public Double rate;
290     }
291     public class cls_returnsFields {
292         public cls_returnsCodeFields[] returnsCodeFields;
293         public cls_returnsDateFields[] returnsDateFields;
294         public cls_returnsIndicatorFields[] returnsIndicatorFi
295         public cls_returnsNumericFields[] returnsNumericFields
296     }
297     public class cls_returnsCodeFields {
298         public String name;
299         public String value;
300     }
301     public class cls_returnsDateFields {
302         public String name;
303         public String value;
304     }
305     public class cls_returnsIndicatorFields {
306         public String name;
307         public boolean value;
308     }
309     public class cls_returnsNumericFields {
310         public String name;
311         public Integer value;
312     }
313     public class cls_seller {
314         public cls_administrativeOrigin administrativeOrigin;
315         public String company;
316         public String department;
317         public cls_dispatcher dispatcher;
318         public String division;
319         public boolean nexusIndicator;
320         public String nexusReasonCode;
321         public cls_physicalOrigin physicalOrigin;
322         public cls_taxRegistrations[] taxRegistrations;
323         public String utilityProvider;
324     }
325     public class cls_administrativeOrigin {
326         public String city;
327         public String country;
328         public cls_currencyConversion currencyConversion;
329         public String externalJurisdictionCode;
330         public String latitude;
331         public String locationCode;
332         public String locationCustomsStatus;
333         public String longitude;
334         public String mainDivision;
335         public String postalCode;
336         public String streetAddress1;
337         public String streetAddress2;
338         public String subDivision;
339         public String taxAreaId;
340     }
341     public class cls_dispatcher {
342         public cls_dispatcherCode dispatcherCode;
343         public cls_taxRegistrations[] taxRegistrations;
344     }
345     public class cls_dispatcherCode {
346         public String classCode;

```

		<pre> 347 public String value; 348 } 349 public class cls_physicalOrigin { 350 public String city; 351 public String country; 352 public cls_currencyConversion currencyConversion; 353 public String externalJurisdictionCode; 354 public String latitude; 355 public String locationCode; 356 public String locationCustomsStatus; 357 public String longitude; 358 public String mainDivision; 359 public String postalCode; 360 public String streetAddress1; 361 public String streetAddress2; 362 public String subDivision; 363 public String taxAreaId; 364 } 365 public class cls_situsOverride { 366 public String taxingLocation; 367 } 368 public class cls_statisticalValue { 369 public Integer amount; 370 public cls_currency currency_z; 371 } 372 public class cls_supplementaryUnit { 373 public String unitType; 374 public Integer value; 375 } 376 public class cls_taxOverride { 377 public String overrideReasonCode; 378 public String overrideType; 379 } 380 public class cls_volume { 381 public String unitOfMeasure; 382 public Double value; 383 } 384 public class cls_weight { 385 public String unitOfMeasure; 386 public Double value; 387 } 388 public class cls_originalCurrency { 389 public String isoCurrencyCodeAlpha; 390 public Integer isoCurrencyCodeNum; 391 public String isoCurrencyName; 392 } 393 public static VertexTaxRequest parse(String json){ 394 return (VertexTaxRequest) System.JSON.deserialize(json 395 } 396 }</pre>
VertexTaxResponse	This class is a DTO used to parse tax response returned by Vertex	<pre> 1 public class VertexTaxResponse{ 2 public cls_data data; 3 public cls_meta meta; 4 public class cls_data { 5 public cls_currency currency_z; 6 public cls_customer customer; 7 public String documentDate; 8 public String documentNumber; 9 public cls_lineItems[] lineItems;</pre>

```

10     public String postingDate;
11     public boolean returnAssistedParametersIndicator;
12     public boolean roundAtLineLevel;
13     public String saleMessageType;
14     public cls_seller seller;
15     public Double subTotal;
16     public String taxPointDate;
17     public Double total;
18     public Double totalTax;
19     public String transactionId;
20     public String transactionType;
21 }
22 public class cls_currency {
23     public String isoCurrencyCodeAlpha;
24     public Integer isoCurrencyCodeNum;
25     public String isoCurrencyName;
26 }
27 public class cls_customer {
28     public cls_administrativeDestination administrativeDes
29     public cls_customerCode customerCode;
30     public cls_destination destination;
31     public boolean isTaxExempt;
32     public cls_taxRegistrations[] taxRegistrations;
33 }
34 public class cls_administrativeDestination {
35     public String city;
36     public String country;
37     public String mainDivision;
38     public String postalCode;
39     public String streetAddress1;
40     public String taxAreaId;
41 }
42 public class cls_customerCode {
43     public String classCode;
44     public boolean isBusinessIndicator;
45     public String value;
46 }
47 public class cls_destination {
48     public String city;
49     public String country;
50     public String mainDivision;
51     public String postalCode;
52     public String streetAddress1;
53     public String taxAreaId;
54 }
55 public class cls_taxRegistrations {
56 }
57 public class cls_lineItems {
58     public cls_customer customer;
59     public Double extendedPrice;
60     public Double fairMarketValue;
61     public String lineItemId;
62     public Integer lineItemNumber;
63     public cls_product product;
64     public cls_quantity quantity;
65     public cls_seller seller;
66     public boolean taxIncludedIndicator;
67     public cls_taxes[] taxes;
68     public Double totalTax;
69     public String transactionType;

```

```
70         public String usage;
71     }
72     public class cls_product {
73         public String productClass;
74         public String value;
75     }
76     public class cls_quantity {
77         public Double value;
78     }
79     public class cls_seller {
80         public cls_administrativeOrigin administrativeOrigin;
81         public String company;
82         public cls_physicalOrigin physicalOrigin;
83         public cls_taxRegistrations[] taxRegistrations;
84     }
85     public class cls_administrativeOrigin {
86         public String city;
87         public String country;
88         public String mainDivision;
89         public String postalCode;
90         public String streetAddress1;
91         public String taxAreaId;
92     }
93     public class cls_physicalOrigin {
94         public String city;
95         public String country;
96         public String mainDivision;
97         public String postalCode;
98         public String streetAddress1;
99         public String taxAreaId;
100    }
101    public class cls_taxes {
102        public Double calculatedTax;
103        public cls_calculationRuleId calculationRuleId;
104        public Double effectiveRate;
105        public Double exempt;
106        public cls_imposition imposition;
107        public cls_impositionType impositionType;
108        public boolean isService;
109        public cls_jurisdiction jurisdiction;
110        public boolean maxTaxIndicator;
111        public Double nominalRate;
112        public Double nonTaxable;
113        public boolean notRegisteredIndicator;
114        public String situs;
115        public String taxCollectedFromParty;
116        public String taxResult;
117        public String taxStructure;
118        public String taxType;
119        public Double taxable;
120    }
121    public class cls_calculationRuleId {
122        public boolean salesTaxHolidayIndicator;
123        public boolean userDefined;
124        public String value;
125    }
126    public class cls_imposition {
127        public String impositionId;
128        public boolean userDefined;
129        public String value;
```

```
130     }
131     public class cls_impositionType {
132         public String impositionTypeId;
133         public boolean userDefined;
134         public String value;
135     }
136     public class cls_jurisdiction {
137         public String effectiveDate;
138         public String expirationDate;
139         public String jurisdictionId;
140         public String jurisdictionType;
141         public String value;
142     }
143     public class cls_meta {
144         public String app;
145         public Integer timeElapsed;
146         public String timeReceived;
147     }
148     public static VertexTaxResponse parse(String json){
149         return (VertexTaxResponse) System.JSON.deserialize(json)
150     }
151 }
```