SF-LWR Extension Framework : UPS Shipping Integration

Functional Overview

This document provides details specific to integrating with UPS.

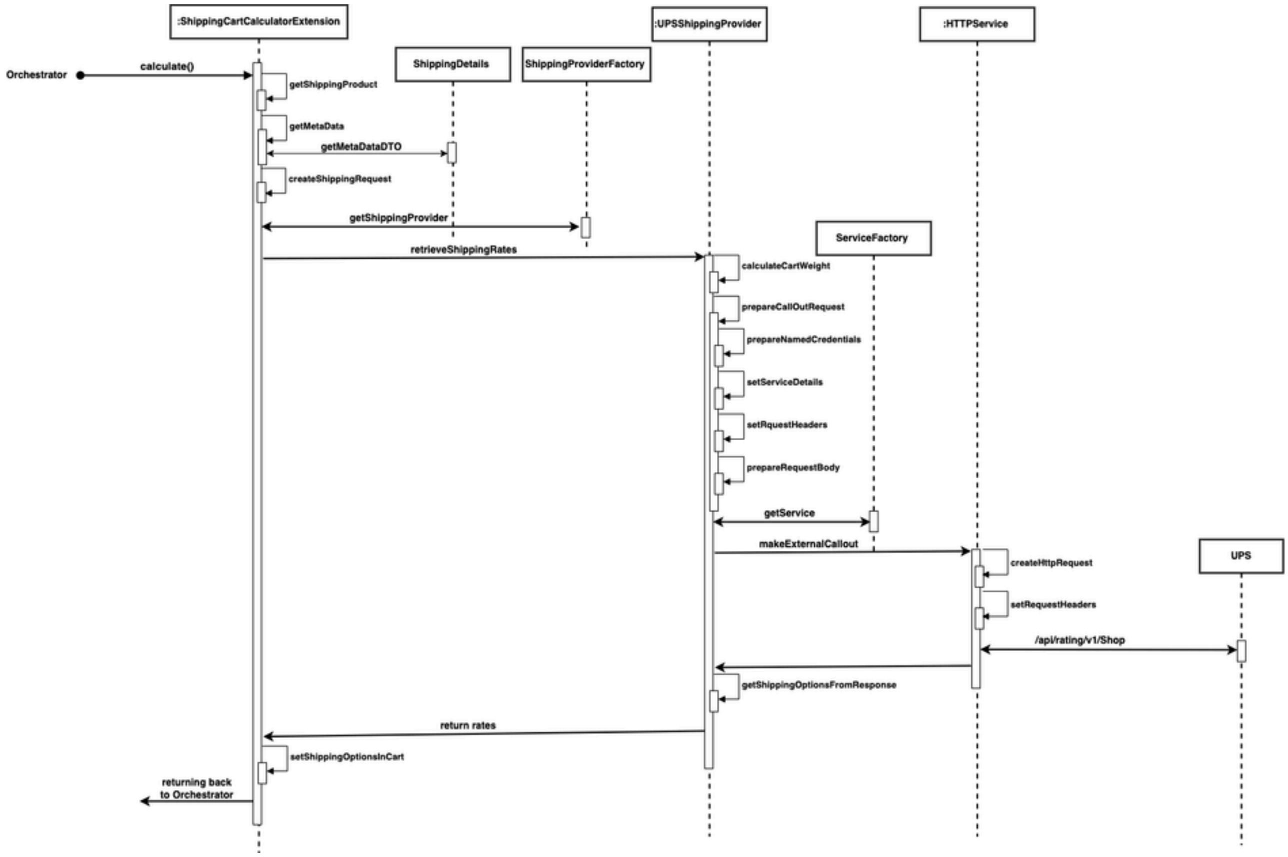
For common details you can refer to parent doc here.

Technical Details

This integration implementation has two parts :

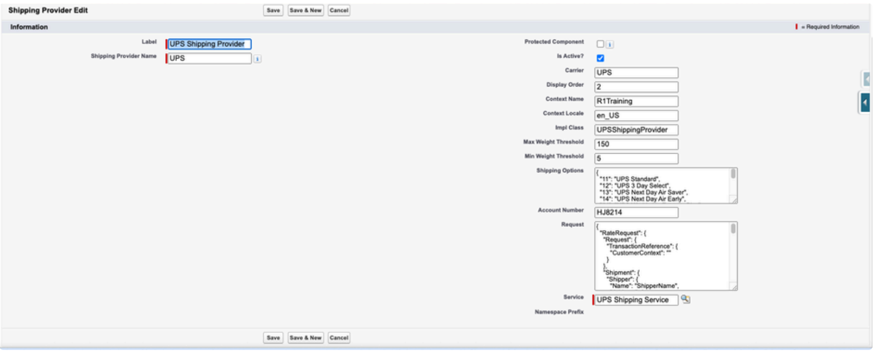
- **Configuration**
  - Custom Meta Data Records
    - UPS Shipping Provider
    - UPS Shipping Service
  - Named Credentials
    - UPS Shipping
  - External Credentials
    - UPS Authentication
    - UPSCredentials (Principal)
- **Source Code**
  - UPSShippingProvider
  - UPSShippingRateRequest
  - UPSShippingRateResponse

Sequence Diagram

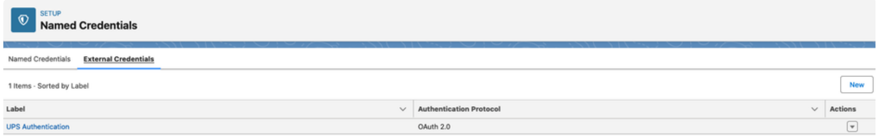


Configuration details

	Section	Task	Details
1	Salesforce Org	Shipping Provider - Create a record for UPS	Create a record **UPS Shipping** for Custom Meta Data Type Shipping Provider

2	Salesforce Org	HTTP Service - Create a record for UPS	Create a record - UPS Shipping of above Custom Meta Data Type
			
			Shipping Provider Edit
			Save · Save & New · Cancel
			Information · ❙ = Required Information
			Label: UPS Shipping Provider
			Shipping Provider Name: UPS
			Protected Component: □
			Is Active?: ☑
			Carrier: UPS
			Display Order: 2
			Context Name: R1Training
			Context Locale: en_US
			Impl Class: UPSShippingProvider
			Max Weight Threshold: 150
			Min Weight Threshold: 5
			Shipping Options: {
			"11": "UPS Standard",
			"12": "UPS 3 Day Select",
			"13": "UPS Next Day Air Saver",
			"14": "UPS Next Day Air Early",
			Account Number: HJ8214
			Request: {
			"RateRequest": {
			"Request": {
			"TransactionReference": {
			"CustomerContext": ""
			}
			},
			"Shipment": {
			"Shipper": {
			"Name": "ShipperName",
			Service: UPS Shipping Service
			Namespace Prefix
			Save · Save & New · Cancel

3	Salesforce Org	Create External Credentials	Create **External Credentials** for OAuth 2.0,



SETUP

Named Credentials

Named Credentials | External Credentials

1 Items - Sorted by Label

Label	Authentication Protocol	Actions
UPS Authentication	OAuth 2.0	▾

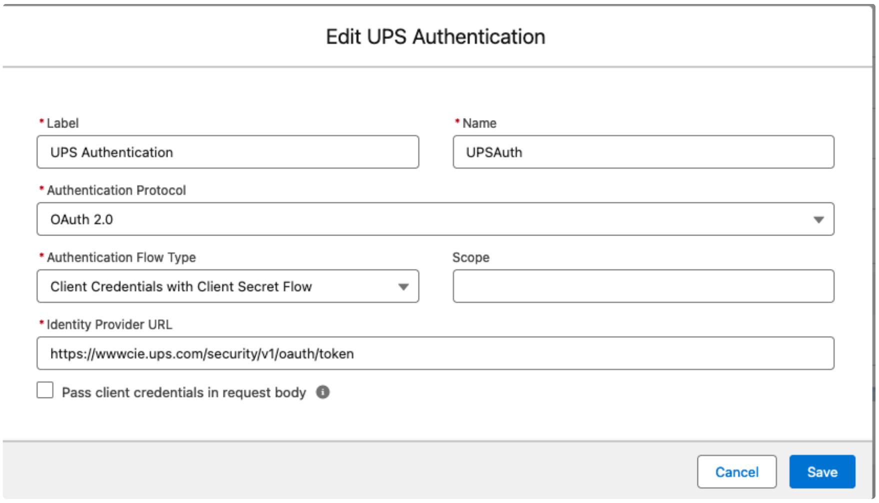
Provide details , like

Name - UPSAuth

Authentication Protocol - AOuth 2.0

Authentication Flow Type - Client Credentials with Secret Flow

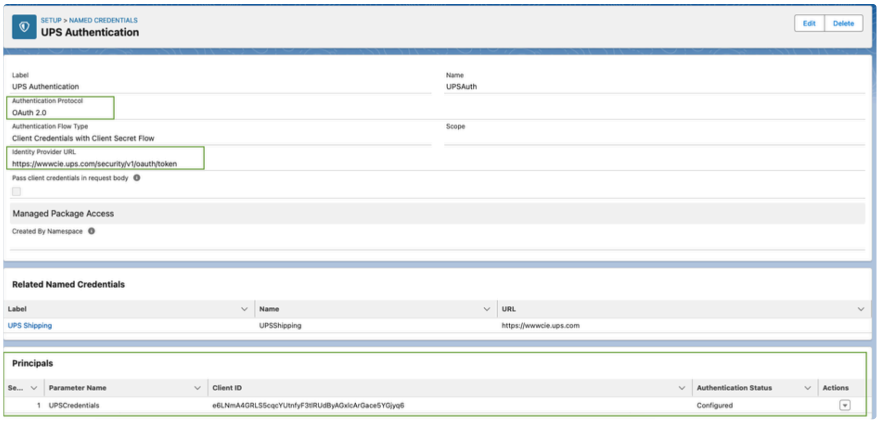
Identity Provider URL - https://wwwcie.ups.com/security/v1/oauth/token



Edit UPS Authentication

* Label: UPS Authentication	* Name: UPSAuth
* Authentication Protocol: OAuth 2.0	
* Authentication Flow Type: Client Credentials with Client Secret Flow	Scope:
* Identity Provider URL: https://wwwcie.ups.com/security/v1/oauth/token	

□ Pass client credentials in request body



SETUP > NAMED CREDENTIALS

UPS Authentication

Label: UPS Authentication	Name: UPSAuth
Authentication Protocol: OAuth 2.0	
Authentication Flow Type: Client Credentials with Client Secret Flow	Scope:
Identity Provider URL: https://wwwcie.ups.com/security/v1/oauth/token	

Pass client credentials in request body □

Managed Package Access

Created By Namespace

Related Named Credentials

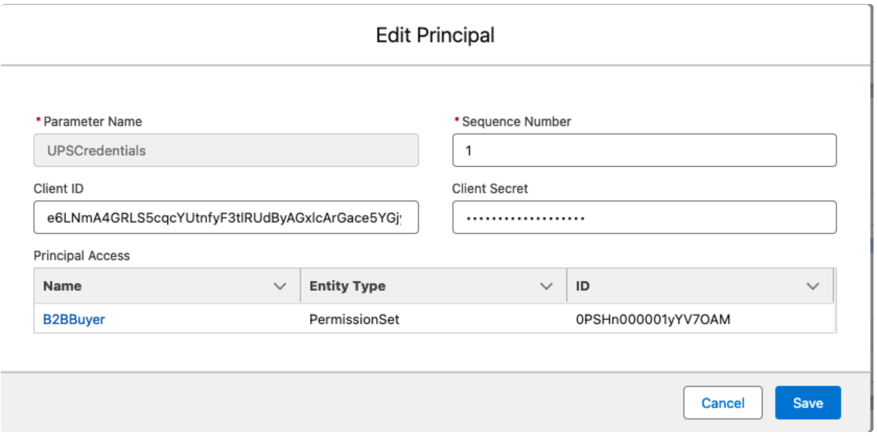
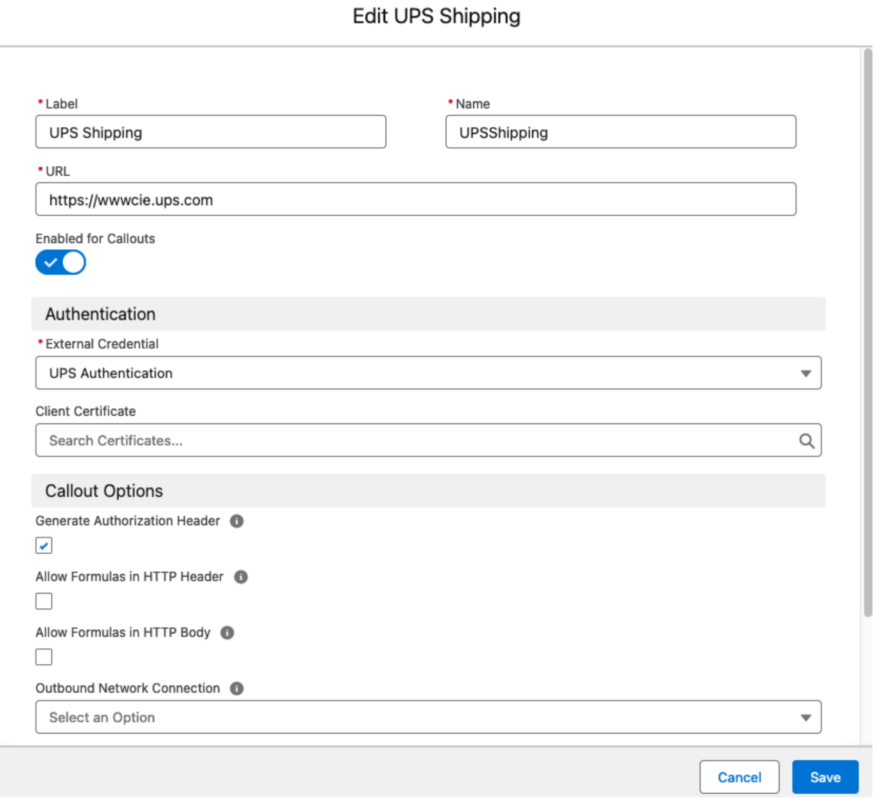
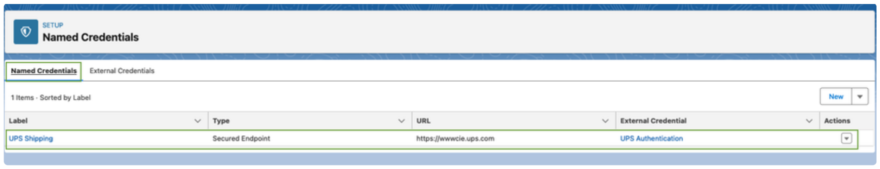
Label	Name	URL
UPS Shipping	UPSShipping	https://wwwcie.ups.com

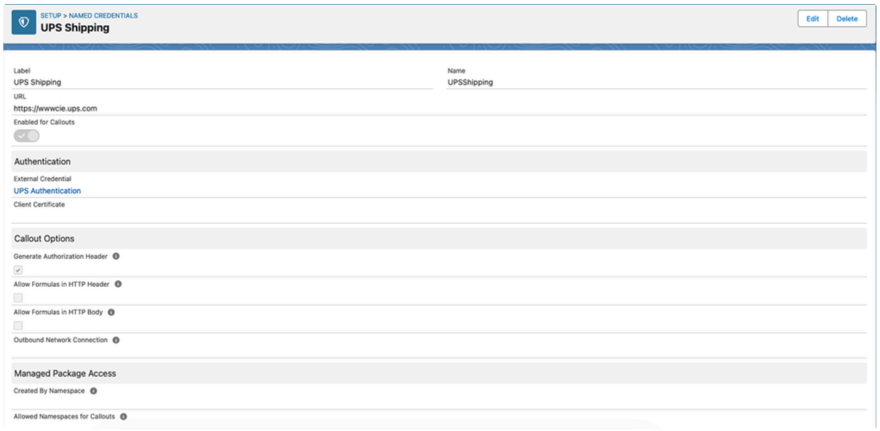
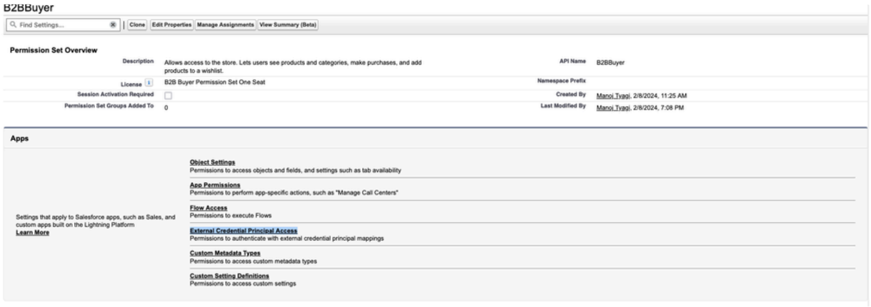
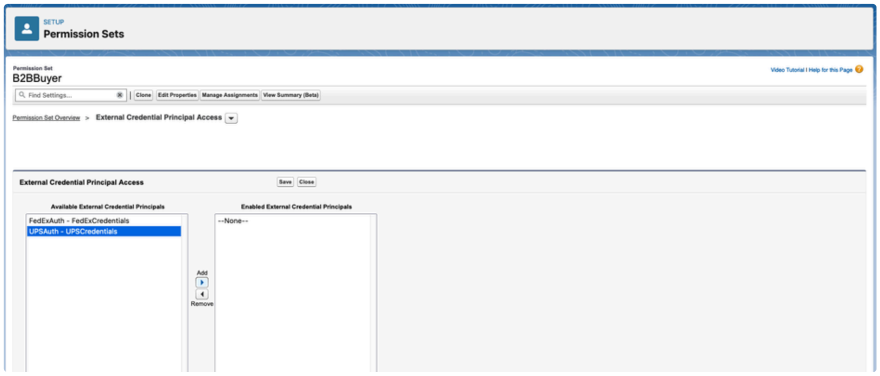
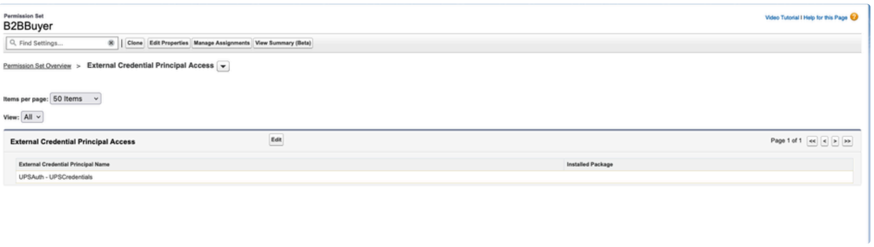
Principals

Se...	Parameter Name	Client ID	Authentication Status	Actions
1	UPSCredentials	e6LNmA4GRLS5cqcYUtnfyF3tlRUdByAGxlcArGace5YGjyq6	Configured	▾

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

Name - UPSCredentials

			 Edit Principal * Parameter Name: UPSCredentials * Sequence Number: 1 Client ID: e6LNmA4GRLS5cqcYUtnfyF3tlRUdByAGxlcArGace5YGj Client Secret: •••••••••••••••••••• Principal Access Name | Entity Type | ID B2BBuyer | PermissionSet | 0PSHn000001yYV7OAM
4	Salesforce Org	Create Named Credentials	Create a Named Credentials for storing UPS API URL Set URL - https://wwwcie.ups.com https://onlinetools.ups.com Also to associate with External Credentials (created in previous step) select “UPS Credentials” under External Credentials Under **Callout Options** Turn on - Generate Authorization Header  Edit UPS Shipping * Label: UPS Shipping * Name: UPSShipping * URL: https://wwwcie.ups.com Enabled for Callouts: ✓ Authentication * External Credential: UPS Authentication Client Certificate: Search Certificates... Callout Options Generate Authorization Header ☑ Allow Formulas in HTTP Header ☐ Allow Formulas in HTTP Body ☐ Outbound Network Connection: Select an Option  SETUP Named Credentials 1 Items · Sorted by Label Label | Type | URL | External Credential | Actions UPS Shipping | Secured Endpoint | https://wwwcie.ups.com | UPS Authentication | ▾

			Here are the details  UPS Shipping Label: UPS Shipping Name: UPSShipping URL: https://wwwcie.ups.com Enabled for Callouts Authentication External Credential: UPS Authentication 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**  Find Settings... Clone Edit Properties Manage Assignments View Summary (Beta) B2BBuyer 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 License: B2B Buyer Permission Set One Seat Namespace Prefix Session Activation Required ☐ Created By: Manoj Tyagi, 2/8/2024, 11:25 AM Permission Set Groups Added To: 0 Last Modified By: Manoj Tyagi, 2/8/2024, 7:08 PM Apps Settings that apply to Salesforce apps, such as Sales, and custom apps built on the Lightning Platform Learn More 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 Now click edit and select principal created for UPS in above steps(step # 3)  SETUP Video Tutorial I Help for this Page Find Settings... Clone Edit Properties Manage Assignments View Summary (Beta) Save Close Add Remove Permission Sets Permission Set: B2BBuyer Permission Set Overview > External Credential Principal Access External Credential Principal Access Available External Credential Principals: FedExAuth - FedExCredentials; UPSAuth - UPSCredentials Enabled External Credential Principals: --None--  Video Tutorial I Help for this Page Find Settings... Clone Edit Properties Manage Assignments View Summary (Beta) Edit Page 1 of 1 << < > >> Permission Set: B2BBuyer Permission Set Overview > External Credential Principal Access Items per page: 50 Items View: All External Credential Principal Access External Credential Principal Name: UPSAuth - UPSCredentials Installed Package

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

Record - UPS Shipping (Shipping Provider **)**

<table>
<tr><th>Field Label</th><th>Field Value</th></tr>
<tr><td>Context Name</td><td>&lt;Webstore name> e.g. R1Training</td></tr>
<tr><td>Context Locale</td><td>&lt;locale> e.g. en_US</td></tr>
<tr><td>Impl Class</td><td>UPSShippingProvider</td></tr>
<tr><td>Max Weight Threshold</td><td>150</td></tr>
<tr><td>Min Weight Threshold</td><td>5</td></tr>
<tr><td>Service</td><td>UPS Shipping Service</td></tr>
<tr><td>Account Number</td><td>&lt;account number > provider by shipping provider with other credentials</td></tr>
<tr><td>Shipping Options</td><td><pre>
{
  "11": "UPS Standard",
  "12": "UPS 3 Day Select",
  "13": "UPS Next Day Air Saver",
  "14": "UPS Next Day Air Early",
  "54": "UPS Worldwide Express Plus",
  "59": "UPS 2nd Day Air A.M.",
  "65": "UPS Worldwide Saver",
  "70": "UPS Access Point Economy",
  "82": "UPS Today Standard",
  "83": "UPS Today Dedicated Courier",
  "84": "UPS Today Intercity",
  "85": "UPS Today Express",
  "86": "UPS Today Express Saver",
  "96": "UPS Worldwide Express Freight",
  "01": "UPS Next Day Air",
  "02": "UPS 2nd Day Air",
  "03": "UPS Ground ",
  "07": "UPS Worldwide Express",
  "08": "UPS Worldwide Expedited"
}
</pre></td></tr>
<tr><td>Request</td><td><pre>
{
  "RateRequest": {
    "Request": {
      "TransactionReference": {
        "CustomerContext": ""
      }
    },
    "Shipment": {
      "Shipper": {
        "Name": "ShipperName",
        "Address": {
          "AddressLine": [
            "ShipperAddressLine"
          ],
</pre></td></tr>
</table>

```
          "City": "TIMONIUM",
          "StateProvinceCode": "MD",
          "PostalCode": "21093",
          "CountryCode": "US"
        }
      },
      "ShipTo": {},
      "ShipFrom": {
        "Name": "ShipFromName",
        "Address": {
          "AddressLine": [
            "ShipFromAddressLine"
          ],
          "City": "TIMONIUM",
          "StateProvinceCode": "MD",
          "PostalCode": "21093",
          "CountryCode": "US"
        }
      },
      "Package": [{
        "PackagingType": {
          "Code": "02",
          "Description": "Packaging"
        },
        "PackageWeight": {
          "UnitOfMeasurement": {
            "Code": "LBS",
            "Description": "Pounds"
          },
          "Weight": "0"
        }
      }]
    }
  }
}
```

Record - UPS Shipping (HTTP Service**)**

Field Label	**Field Value**
End Point	/api/rating/v1/Shop
HTTP Method	GET POST PUT DELETE PATCH HEAD TRACE
Impl Class	*This is optional so could be left blank*
Named Credentials	UPSShipping
Service Mode	Live

	Mocked
Timeout	5000 (ms)
Success Codes	200,201
Error Codes	400,403,404,500,503
Auth Codes	401

Mocked Response

```
{
    "RateResponse": {
        "Response": {
            "ResponseStatus": {
                "Code": "1",
                "Description": "Success"
            },
            "Alert": [
                {
                    "Code": "119001",
                    "Description": "Additional Handling has automatically been set on Package 1."
                },
                {
                    "Code": "110971",
                    "Description": "Your invoice may vary from the displayed reference rates"
                }
            ],
            "TransactionReference": {
                "CustomerContext": "CustomerContext"
            }
        },
        "RatedShipment": [
            {
                "Service": {
                    "Code": "01",
                    "Description": ""
                },
                "RatedShipmentAlert": [
                    {
                        "Code": "119001",
                        "Description": "Additional Handling has automatically been set on Package 1."
                    },
                    {
                        "Code": "110971",
                        "Description": "Your invoice may vary from the displayed reference rates"
                    }
                ],
                "BillingWeight": {
                    "UnitOfMeasurement": {
                        "Code": "LBS",
                        "Description": "Pounds"
                    },
                    "Weight": "150.0"
                },
        "TransportationCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "1253.30"
        },
        "ServiceOptionsCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "41.00"
        },
        "TotalCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "1294.30"
        },
        "GuaranteedDelivery": {
            "BusinessDaysInTransit": "1",
            "DeliveryByTime": "10:30 A.M."
        },
        "RatedPackage": {
            "TransportationCharges": {
                "CurrencyCode": "USD",
                "MonetaryValue": "1253.30"
            },
            "ServiceOptionsCharges": {
                "CurrencyCode": "USD",
                "MonetaryValue": "41.00"
            },
            "TotalCharges": {
                "CurrencyCode": "USD",
                "MonetaryValue": "1294.30"
            },
            "Weight": "150.0",
            "BillingWeight": {
                "UnitOfMeasurement": {
                    "Code": "LBS",
                    "Description": "Pounds"
                },
                "Weight": "150.0"
            }
        }
    },
    {
        "Service": {
            "Code": "02",
            "Description": ""
        },
        "RatedShipmentAlert": [
            {
                "Code": "119001",
                "Description": "Additional Handling has automatically been set on Package 1."
            },
            {
                "Code": "110971",
                "Description": "Your invoice may vary from the displayed reference rates"
        }
    ],
    "BillingWeight": {
        "UnitOfMeasurement": {
            "Code": "LBS",
            "Description": "Pounds"
        },
        "Weight": "150.0"
    },
    "TransportationCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "520.06"
    },
    "ServiceOptionsCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "41.00"
    },
    "TotalCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "561.06"
    },
    "GuaranteedDelivery": {
        "BusinessDaysInTransit": "2"
    },
    "RatedPackage": {
        "TransportationCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "520.06"
        },
        "ServiceOptionsCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "41.00"
        },
        "TotalCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "561.06"
        },
        "Weight": "150.0",
        "BillingWeight": {
            "UnitOfMeasurement": {
                "Code": "LBS",
                "Description": "Pounds"
            },
            "Weight": "150.0"
        }
    }
},
{
    "Service": {
        "Code": "03",
        "Description": ""
    },
    "RatedShipmentAlert": [
        {
            "Code": "119001",
            "Description": "Additional Handling has automatically been set on Package 1."
        },
        {
            "Code": "110971",
            "Description": "Your invoice may vary from the displayed reference rates"
        }
    ],
    "BillingWeight": {
        "UnitOfMeasurement": {
            "Code": "LBS",
            "Description": "Pounds"
        },
        "Weight": "150.0"
    },
    "TransportationCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "144.14"
    },
    "ServiceOptionsCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "41.00"
    },
    "TotalCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "185.14"
    },
    "RatedPackage": {
        "TransportationCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "144.14"
        },
        "ServiceOptionsCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "41.00"
        },
        "TotalCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "185.14"
        },
        "Weight": "150.0",
        "BillingWeight": {
            "UnitOfMeasurement": {
                "Code": "LBS",
                "Description": "Pounds"
            },
            "Weight": "150.0"
        }
    }
},
{
    "Service": {
        "Code": "12",
        "Description": ""
    },
    "RatedShipmentAlert": [
        {
            "Code": "119001",
            "Description": "Additional Handling has automatically been set on Package 1."
        },
        {
            "Code": "110971",
            "Description": "Your invoice may vary from the displayed reference rates"
        }
    ],
    "BillingWeight": {
        "UnitOfMeasurement": {
            "Code": "LBS",
            "Description": "Pounds"
        },
        "Weight": "150.0"
    },
    "TransportationCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "368.79"
    },
    "ServiceOptionsCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "41.00"
    },
    "TotalCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "409.79"
    },
    "GuaranteedDelivery": {
        "BusinessDaysInTransit": "3"
    },
    "RatedPackage": {
        "TransportationCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "368.79"
        },
        "ServiceOptionsCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "41.00"
        },
        "TotalCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "409.79"
        },
        "Weight": "150.0",
        "BillingWeight": {
            "UnitOfMeasurement": {
                "Code": "LBS",
                "Description": "Pounds"
            },
            "Weight": "150.0"
        }
    }
},
{
    "Service": {
        "Code": "13",
        "Description": ""
    },
    "RatedShipmentAlert": [
        {
            "Code": "119001",
            "Description": "Additional Handling has automatically been set on Package 1."
        },
        {
            "Code": "110971",
            "Description": "Your invoice may vary from the displayed reference rates"
        }
    ],
    "BillingWeight": {
        "UnitOfMeasurement": {
            "Code": "LBS",
            "Description": "Pounds"
        },
        "Weight": "150.0"
    },
    "TransportationCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "1226.58"
    },
    "ServiceOptionsCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "41.00"
    },
    "TotalCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "1267.58"
    },
    "GuaranteedDelivery": {
        "BusinessDaysInTransit": "1",
        "DeliveryByTime": "3:00 P.M."
    },
    "RatedPackage": {
        "TransportationCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "1226.58"
        },
        "ServiceOptionsCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "41.00"
        },
        "TotalCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "1267.58"
        },
        "Weight": "150.0",
        "BillingWeight": {
            "UnitOfMeasurement": {
                "Code": "LBS",
                "Description": "Pounds"
            },
            "Weight": "150.0"
        }
    }
},
{
    "Service": {
        "Code": "14",
        "Description": ""
    },
    "RatedShipmentAlert": [
        {
            "Code": "119001",
            "Description": "Additional Handling has automatically been set on Package 1."
        },
        {
            "Code": "110971",
            "Description": "Your invoice may vary from the displayed reference rates"
        }
    ],
    "BillingWeight": {
        "UnitOfMeasurement": {
            "Code": "LBS",
            "Description": "Pounds"
        },
        "Weight": "150.0"
    },
    "TransportationCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "1288.55"
    },
    "ServiceOptionsCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "41.00"
    },
    "TotalCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "1329.55"
    },
    "GuaranteedDelivery": {
        "BusinessDaysInTransit": "1",
        "DeliveryByTime": "8:30 A.M."
    },
    "RatedPackage": {
        "TransportationCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "1288.55"
        },
        "ServiceOptionsCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "41.00"
        },
        "TotalCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "1329.55"
        },
        "Weight": "150.0",
        "BillingWeight": {
            "UnitOfMeasurement": {
                "Code": "LBS",
                "Description": "Pounds"
            },
            "Weight": "150.0"
        }
    }
},
{
    "Service": {
        "Code": "59",
        "Description": ""
    },
    "RatedShipmentAlert": [
        {
            "Code": "119001",
            "Description": "Additional Handling has automatically been set on Package 1."
        },
        {
            "Code": "110971",
            "Description": "Your invoice may vary from the displayed reference rates"
        }
    ],
    "BillingWeight": {
        "UnitOfMeasurement": {
            "Code": "LBS",
            "Description": "Pounds"
        },
        "Weight": "150.0"
    },
    "TransportationCharges": {
        "CurrencyCode": "USD",
        "MonetaryValue": "634.06"
    },
        "ServiceOptionsCharges": {
          "CurrencyCode": "USD",
          "MonetaryValue": "41.00"
        },
        "TotalCharges": {
          "CurrencyCode": "USD",
          "MonetaryValue": "675.06"
        },
        "GuaranteedDelivery": {
          "BusinessDaysInTransit": "2",
          "DeliveryByTime": "10:30 A.M."
        },
        "RatedPackage": {
          "TransportationCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "634.06"
          },
          "ServiceOptionsCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "41.00"
          },
          "TotalCharges": {
            "CurrencyCode": "USD",
            "MonetaryValue": "675.06"
          },
          "Weight": "150.0",
          "BillingWeight": {
            "UnitOfMeasurement": {
              "Code": "LBS",
              "Description": "Pounds"
            },
            "Weight": "150.0"
          }
        }
      }
    ]
  }
}
```

Source Code

<table>
<tr><th>Class</th><th>Details</th><th>Code</th></tr>
<tr><td>UPSShippingProvider</td><td>This class extends ShippingProvider class & implements following methods -
<b>prepareRequestBody</b>
<b>getShippingOptionsFromResponse</b></td><td><pre>public class UPSShippingProvider extends ShippingProvider {
  public UPSShippingProvider() {
  }
  public virtual override void prepareRequestBody(
    ShippingProviderRequest shippingRequest,
    Map<String, String> callOutRequest
) {
    UPSShippingRateRequest requestObject = UPSShippingRateReque
      shippingRequest.shippingMetaData.requestJSON
    );</pre></td></tr>
</table>

```
    if (requestObject != null) {
      requestObject.RateRequest.Shipment.Shipper.ShipperNumber
      requestObject.RateRequest.Shipment.ShipTo = setShipToAddr
        shippingRequest
      );
      requestObject.RateRequest.Shipment.Package_Z[0]
        .PackageWeight.Weight = String.valueOf(shippingRequest.
      String strRatingRequestBody = JSON.serialize(requestObjec
      strRatingRequestBody = strRatingRequestBody.replaceAll(
          'Package_Z',
          'Package'
        )
        .replaceAll('Date_Z', 'Date')
        .replaceAll('Time_Z', 'Time');

      callOutRequest.put(Constants.SERVICE_REQUEST_BODY, strRat
    }
  }

  public virtual UPSShippingRateRequest.ShipTo setShipToAddress
    ShippingProviderRequest shippingRequest
  ) {
    UPSShippingRateRequest.Address address = new UPSShippingRat
    address.AddressLine = new List<String>();
    address.AddressLine.add(shippingRequest.street);
    address.City = shippingRequest.city;
    address.StateProvinceCode = shippingRequest.state;
    address.PostalCode = shippingRequest.postalCode;
    address.CountryCode = shippingRequest.country;
    UPSShippingRateRequest.ShipTo shipTo = new UPSShippingRateR
    shipTo.Address = address;
    return shipTo;
  }

  public virtual override Map<String, ShippingProviderResponse>
    List<String> responseList,
    ShippingProviderRequest shippingRequest
  ) {
    Map<String, Object> shippingMethods = new Map<String, Objec
    Map<String, ShippingProviderResponse> shippingMethodsWithRa
      responseList,
      shippingRequest
    );
    return shippingMethodsWithRate;
  }

  public virtual Map<String, ShippingProviderResponse> parseUPS
    List<String> lstUPSResponseBody,
    ShippingProviderRequest shippingRequest
  ) {
    Map<String, ShippingProviderResponse> shippingMethodsWithRa
    ShippingProviderResponse shippingResponse;
    if (lstUPSResponseBody != null && lstUPSResponseBody.size()
      for (String strUPSResponseBody : lstUPSResponseBody) {
        UPSShippingRateResponse upsShippingRateResponseObj = UP
          strUPSResponseBody
        );
        UPSShippingRateResponse.RateResponse upsShippingRateRes
        for (
          UPSShippingRateResponse.RatedShipment shipmentObj : u
```

<table>
<tr>
<td></td>
<td></td>
<td><pre>
) {
          Decimal shippingAmount = shipmentObj.TotalCharges.Mon
            null
            ? Decimal.valueOf(shipmentObj.TotalCharges.Monetary
            : 0;
          if (shippingMethodsWithRate.containsKey(shipmentObj.S
            shippingResponse = shippingMethodsWithRate.get(
              shipmentObj.Service.Code
            );
            shippingResponse.cost = shippingResponse.cost + shi
          } else {
            shippingResponse = new ShippingProviderResponse();
            shippingResponse.serviceCode = shipmentObj.Service.
            shippingResponse.cost = shippingAmount;
            shippingResponse.carrier = shippingRequest.shipping
            shippingResponse.displayOrder = shippingRequest.shi
            shippingMethodsWithRate.put(
              shipmentObj.Service.Code,
              shippingResponse
            );
          }
        }
      }
    }
    return shippingMethodsWithRate;
  }
}
</pre></td>
</tr>
<tr>
<td>UPSShippingRateRequest</td>
<td>This class is a DTO used to prepare UPS Shipping rate request which is posted to UPS to get shipping details</td>
<td><pre>

public class UPSShippingRateRequest {

    public class TransactionReference {
        public String CustomerContext {get;set;}

        public TransactionReference(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'CustomerContext') {
                            CustomerContext = parser.getText()
                        } else {
                            System.debug(LoggingLevel.WARN, 'T
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public RateRequest RateRequest {get;set;}

    public UPSShippingRateRequest(JSONParser parser) {
        while (parser.nextToken() != System.JSONToken.END_OBJE
            if (parser.getCurrentToken() == System.JSONToken.F
                String text = parser.getText();
                if (parser.nextToken() != System.JSONToken.VAL
                    if (text == 'RateRequest') {
                        RateRequest = new RateRequest(parser);
</pre></td>
</tr>
</table>

```
                    } else {
                        System.debug(LoggingLevel.WARN, 'UPSSh
                        consumeObject(parser);
                    }
                }
            }
        }
    }

    public class Address {
        public List<String> AddressLine {get;set;}
        public String City {get;set;}
        public String StateProvinceCode {get;set;}
        public String PostalCode {get;set;}
        public String CountryCode {get;set;}
    public Address() {}
        public Address(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'AddressLine') {
                            AddressLine = arrayOfString(parser
                        } else if (text == 'City') {
                            City = parser.getText();
                        } else if (text == 'StateProvinceCode'
                            StateProvinceCode = parser.getText
                        } else if (text == 'PostalCode') {
                            PostalCode = parser.getText();
                        } else if (text == 'CountryCode') {
                            CountryCode = parser.getText();
                        } else {
                            System.debug(LoggingLevel.WARN, 'A
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class Pickup {
        public String Date_Z {get;set;} // in json: Date
        public String Time_Z {get;set;} // in json: Time

        public Pickup(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'Date') {
                            Date_Z = parser.getText();
                        } else if (text == 'Time') {
                            Time_Z = parser.getText();
                        } else {
                            System.debug(LoggingLevel.WARN, 'P
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class ShipmentTotalWeight {
        public CustomerClassification UnitOfMeasurement {get;s
        public String Weight {get;set;}

        public ShipmentTotalWeight(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'UnitOfMeasurement') {
                            UnitOfMeasurement = new CustomerCl
                        } else if (text == 'Weight') {
                            Weight = parser.getText();
                        } else {
                            System.debug(LoggingLevel.WARN, 'S
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class ShipmentCharges {
        public BillShipper BillShipper {get;set;}

        public ShipmentCharges(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'BillShipper') {
                            BillShipper = new BillShipper(pars
                        } else {
                            System.debug(LoggingLevel.WARN, 'S
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class Request {
        public TransactionReference TransactionReference {get;

        public Request(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'TransactionReference') {
                            TransactionReference = new Transac
                        } else {
                            System.debug(LoggingLevel.WARN, 'R
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class Dimensions {
        public CustomerClassification UnitOfMeasurement {get;s
        public String Length {get;set;}
        public String Width {get;set;}
        public String Height {get;set;}

        public Dimensions(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'UnitOfMeasurement') {
                            UnitOfMeasurement = new CustomerCl
                        } else if (text == 'Length') {
                            Length = parser.getText();
                        } else if (text == 'Width') {
                            Width = parser.getText();
                        } else if (text == 'Height') {
                            Height = parser.getText();
                        } else {
                            System.debug(LoggingLevel.WARN, 'D
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class Shipment {
        public Shipper Shipper {get;set;}
        public ShipTo ShipTo {get;set;}
        public ShipTo ShipFrom {get;set;}
        public List<PaymentDetails> PaymentDetails {get;set;}
        public ShipmentRatingOptions ShipmentRatingOptions {ge
        public CustomerClassification Service {get;set;}
        public ShipmentTotalWeight ShipmentTotalWeight {get;se
        public String NumOfPieces {get;set;}
        public List<Package_Z> Package_Z {get;set;} // in json
        public DeliveryTimeInformation DeliveryTimeInformation
        public ShipmentCharges ShipmentCharges {get;set;}

        public Shipment(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'Shipper') {
                            Shipper = new Shipper(parser);
                        } else if (text == 'ShipTo') {
                            ShipTo = new ShipTo(parser);
                        } else if (text == 'ShipFrom') {
                            ShipFrom = new ShipTo(parser);
                        } else if (text == 'PaymentDetails') {
                            PaymentDetails = arrayOfPaymentDet
                        } else if (text == 'ShipmentRatingOpti
                            ShipmentRatingOptions = new Shipme
                        } else if (text == 'Service') {
                            Service = new CustomerClassificati
                        } else if (text == 'ShipmentTotalWeigh
                            ShipmentTotalWeight = new Shipment
                        } else if (text == 'NumOfPieces') {
                            NumOfPieces = parser.getText();
                        } else if (text == 'Package') {
                            Package_Z = arrayOfPackage_Z(parse
                        } else if (text == 'DeliveryTimeInform
                            DeliveryTimeInformation = new Deli
                        } else if (text == 'ShipmentCharges')
                            ShipmentCharges = new ShipmentChar
                        } else {
                            System.debug(LoggingLevel.WARN, 'S
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class ShipmentCharge {
        public String Type {get;set;}
        public BillThirdParty BillThirdParty {get;set;}

        public ShipmentCharge(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'Type') {
                            Type = parser.getText();
                        } else if (text == 'BillThirdParty') {
                            BillThirdParty = new BillThirdPart
                        } else {
                            System.debug(LoggingLevel.WARN, 'S
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class CustomerClassification {
        public String Code {get;set;}
        public String Description {get;set;}

        public CustomerClassification(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'Code') {
                            Code = parser.getText();
                        } else if (text == 'Description') {
                            Description = parser.getText();
                        } else {
                            System.debug(LoggingLevel.WARN, 'C
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class BillShipper {
        public String AccountNumber {get;set;}

        public BillShipper(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'AccountNumber') {
                            AccountNumber = parser.getText();
                        } else {
                            System.debug(LoggingLevel.WARN, 'B
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class DeliveryTimeInformation {
        public String PackageBillType {get;set;}
        public Pickup Pickup {get;set;}

        public DeliveryTimeInformation(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'PackageBillType') {
                            PackageBillType = parser.getText()
                        } else if (text == 'Pickup') {
                            Pickup = new Pickup(parser);
                        } else {
                            System.debug(LoggingLevel.WARN, 'D
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class Shipper {
        public String Name {get;set;}
        public String ShipperNumber {get;set;}
        public Address Address {get;set;}

        public Shipper(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'Name') {
                            Name = parser.getText();
                        } else if (text == 'ShipperNumber') {
                            ShipperNumber = parser.getText();
                        } else if (text == 'Address') {
                            Address = new Address(parser);
                        } else {
                            System.debug(LoggingLevel.WARN, 'S
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class ShipTo {
        public String Name {get;set;}
        public Address Address {get;set;}
    public ShipTo() {
    }
        public ShipTo(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'Name') {
                            Name = parser.getText();
                        } else if (text == 'Address') {
                            Address = new Address(parser);
                        } else {
                            System.debug(LoggingLevel.WARN, 'S
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class PaymentDetails {
        public ShipmentCharge ShipmentCharge {get;set;}

        public PaymentDetails(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'ShipmentCharge') {
                            ShipmentCharge = new ShipmentCharg
                        } else {
                            System.debug(LoggingLevel.WARN, 'P
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class Package_Z {
        public CustomerClassification SimpleRate {get;set;}
        public CustomerClassification PackagingType {get;set;}
        public Dimensions Dimensions {get;set;}
        public ShipmentTotalWeight PackageWeight {get;set;}
        public String OversizeIndicator {get;set;}
        public String MinimumBillableWeightIndicator {get;set;

        public Package_Z(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'SimpleRate') {
                            SimpleRate = new CustomerClassific
                        } else if (text == 'PackagingType') {
                            PackagingType = new CustomerClassi
                        } else if (text == 'Dimensions') {
                            Dimensions = new Dimensions(parser
                        } else if (text == 'PackageWeight') {
                            PackageWeight = new ShipmentTotalW
                        } else if (text == 'OversizeIndicator'
                            OversizeIndicator = parser.getText
                        } else if (text == 'MinimumBillableWei
                            MinimumBillableWeightIndicator = p
                        } else {
                            System.debug(LoggingLevel.WARN, 'P
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class RateRequest {
        public Request Request {get;set;}
        public CustomerClassification CustomerClassification {
        public Shipment Shipment {get;set;}

        public RateRequest(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'Request') {
                            Request = new Request(parser);
                        } else if (text == 'CustomerClassifica
                            CustomerClassification = new Custo
                        } else if (text == 'Shipment') {
                            Shipment = new Shipment(parser);
                        } else {
                            System.debug(LoggingLevel.WARN, 'R
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class BillThirdParty {
        public String AttentionName {get;set;}
        public String Name {get;set;}
        public String AccountNumber {get;set;}
        public Address_Z Address {get;set;}

        public BillThirdParty(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'AttentionName') {
                            AttentionName = parser.getText();
                        } else if (text == 'Name') {
                            Name = parser.getText();
                        } else if (text == 'AccountNumber') {
                            AccountNumber = parser.getText();
                        } else if (text == 'Address') {
                            Address = new Address_Z(parser);
                        } else {
                            System.debug(LoggingLevel.WARN, 'B
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class Address_Z {
        public String AddressLine {get;set;}
        public String City {get;set;}
        public String StateProvinceCode {get;set;}
        public String PostalCode {get;set;}
        public String CountryCode {get;set;}

    public Address_Z(){

    }
        public Address_Z(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'AddressLine') {
                            AddressLine = parser.getText();
                        } else if (text == 'City') {
                            City = parser.getText();
                        } else if (text == 'StateProvinceCode'
                            StateProvinceCode = parser.getText
                        } else if (text == 'PostalCode') {
                            PostalCode = parser.getText();
                        } else if (text == 'CountryCode') {
                            CountryCode = parser.getText();
                        } else {
                            System.debug(LoggingLevel.WARN, 'A
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }

    public class ShipmentRatingOptions {
        public String TPFCNegotiatedRatesIndicator {get;set;}
        public String NegotiatedRatesIndicator {get;set;}

        public ShipmentRatingOptions(JSONParser parser) {
            while (parser.nextToken() != System.JSONToken.END_
                if (parser.getCurrentToken() == System.JSONTok
                    String text = parser.getText();
                    if (parser.nextToken() != System.JSONToken
                        if (text == 'TPFCNegotiatedRatesIndica
                            TPFCNegotiatedRatesIndicator = par
                        } else if (text == 'NegotiatedRatesInd
                            NegotiatedRatesIndicator = parser.
                        } else {
                            System.debug(LoggingLevel.WARN, 'S
                            consumeObject(parser);
                        }
                    }
                }
            }
        }
    }


    public static UPSShippingRateRequest parse(String json) {
        System.JSONParser parser = System.JSON.createParser(js
        return new UPSShippingRateRequest(parser);
    }

    public static void consumeObject(System.JSONParser parser)
        Integer depth = 0;
        do {
            System.JSONToken curr = parser.getCurrentToken();
            if (curr == System.JSONToken.START_OBJECT ||
                curr == System.JSONToken.START_ARRAY) {
                depth++;
            } else if (curr == System.JSONToken.END_OBJECT ||
                curr == System.JSONToken.END_ARRAY) {
                depth--;
            }
        } while (depth > 0 && parser.nextToken() != null);
    }





    private static List<Package_Z> arrayOfPackage_Z(System.JSO
        List<Package_Z> res = new List<Package_Z>();
        if (p.getCurrentToken() == null) p.nextToken();
        while (p.nextToken() != System.JSONToken.END_ARRAY) {
            res.add(new Package_Z(p));
        }
        return res;
    }
















    private static List<String> arrayOfString(System.JSONParse
        List<String> res = new List<String>();
        if (p.getCurrentToken() == null) p.nextToken();
        while (p.nextToken() != System.JSONToken.END_ARRAY) {
            res.add(p.getText());
        }
        return res;
    }





    private static List<PaymentDetails> arrayOfPaymentDetails(
        List<PaymentDetails> res = new List<PaymentDetails>();
        if (p.getCurrentToken() == null) p.nextToken();
        while (p.nextToken() != System.JSONToken.END_ARRAY) {
            res.add(new PaymentDetails(p));
        }
        return res;
    }



}

```

UPSShippingRateResponse

This class is a DTO used to parse shipping rate response returned by UPS

```
global with sharing class UPSShippingRateResponse {
  global class Response {
    global ResponseStatus ResponseStatus;
    //global List<ResponseStatus> Alert;
    global TransactionReference TransactionReference;
  }

  global class TransactionReference {
    global String CustomerContext;
  }

  global RateResponse RateResponse;

  global class ResponseStatus {
    global String Code;
    global String Description;
  }

  global class BillingWeight {
    global ResponseStatus UnitOfMeasurement;
    global String Weight;
  }

  global class GuaranteedDelivery {
    global String BusinessDaysInTransit;
  }

  global class RateResponse {
    global Response Response;
    global List<RatedShipment> RatedShipment;
  }

  global class TransportationCharges {
    global String CurrencyCode;
    global String MonetaryValue;
  }

  global class RatedPackage {
    global TransportationCharges TransportationCharges;
    global TransportationCharges ServiceOptionsCharges;
    global TransportationCharges TotalCharges;
    global String Weight;
    global BillingWeight BillingWeight;
  }

  global class GuaranteedDelivery_Z {
    global String BusinessDaysInTransit;
    global String DeliveryByTime;
  }

  global class RatedShipment {
    global ResponseStatus Service;
    //global List<ResponseStatus> RatedShipmentAlert;
    global BillingWeight BillingWeight;
    global TransportationCharges TransportationCharges;
    global TransportationCharges ServiceOptionsCharges;
    global TransportationCharges TotalCharges;
    global RatedPackage RatedPackage;
    global GuaranteedDelivery GuaranteedDelivery;
  }

  global static UPSShippingRateResponse parse(String json) {
    return (UPSShippingRateResponse) System.JSON.deserialize(
      json,
      UPSShippingRateResponse.class
    );
  }
}

```