SF-LWR Extension Framework : Shipping Integration (using OAuth)

Integration/Customization Name	Shipping Integration (using OAuth)
Description / Requirements Abstract	This integration can be used wherever there is a need to fetch internal/flat or external shipping rates. It takes care of backend only any UI change would be specific to the project as per the requirement.
Status	COMPLETE
External References	**Salesforce Commerce Extensions** https://developer.salesforce.com/docs/commerce/salesforce-commerce/guide/extensions.html **UPS** **UPS Developer Portal** • Auth Token - https://developer.ups.com/api/reference?loc=en_US#operation/CreateToken • Rating - https://developer.ups.com/api/reference?loc=en_US#operation/Rate **FedEx** **FedEx Developer Portal** • Auth Token – https://developer.fedex.com/api/en-us/catalog/authorization/v1/docs.html • Rating - https://developer.fedex.com/api/en-us/catalog/rate/v1/docs.html

Overview

This document provides details about how to integrate B2B/D2C commerce with a shipping provider (like UPS, FedEx etc.) to **fetch shipping methods & costs/rates.**

This integration is based on **Salesforce Commerce Extension** framework which was introduced in Winter' 24 release and going forward it is ***recommended***

to use extensions over integrations because they offer more targeted customizations for B2B/D2C store.

Capabilities

- Lot of shipping related information can be pre configured like ,
    - Shipper/Account Number
    - Ship From Address
    - Shipper Address
    - Weight thresholds
    - Shipping Options(shipping methods code & name mapping)
    - Mocked response
- In case a store supports multiple locales/countries , can configure locale specific shipping providers so a store can have multiple shipping providers configured
- Can combine shipping rates from multiple shipping providers (e.g. internal flat rates & UPS rates),
    - On UI, rates can be displayed grouping by carrier (have to customize LWC component)
    - Also can be sorted using display order (have to customize LWC component)

Technical Overview

This integration implementation can be divided in to two parts :

- **Configuration**
    - Custom Meta Data Types
        - Shipping Provider
        - HTTP Service
    - Named Credentials
    - External Credentials
- **Source Code**
    - CartExtension.ShippingCartCalculatorExtension
    - ShippingDetails
    - ShippingMetaData
    - ShippingProviderFactory
    - ShippingProvider
    - ShippingProviderRequest
    - ShippingProviderResponse
    - HTTPService

Extensibility

There are various points where this implementation can be extended to achieve project specific requirement:

- Meta Data
    - Can add more fields to both the meta data types Shipping Provider & HTTP Service
- Code
    - Following classes can be extended , details are available under respective item number
        - ShippingCartCalculatorExtension
        - ShippingProvider
        - HTTPService

Limitations

- Currently HTTS Service supports only JSON (as data format)

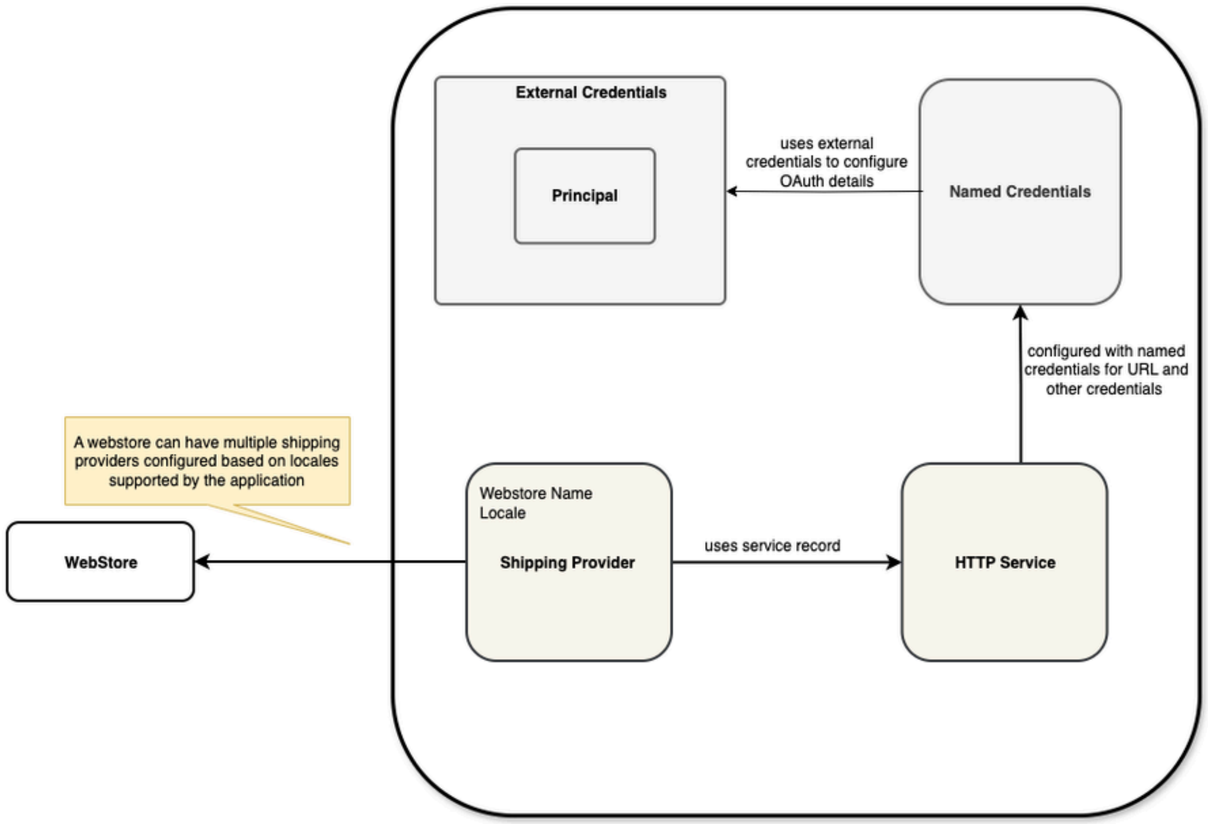
Prerequisites

- Standard object Product2 should have an attribute **Weight__c** to store a product's weight.
- There should be a product with product family configured as **Shipping** (Product2.Family = 'Shipping') , this product will be used as shipping product

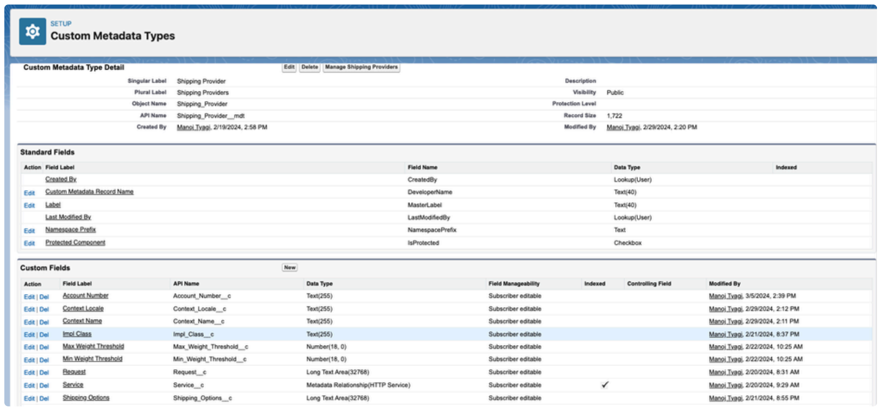
Upcoming Features

- Logging enhancements
- HTTP Service - Support for XML data format
- HTTP Service - Implementing Retry logic (in case callout fails)
- *Anything else as per feedback*

Configuration


External Credentials
Principal
uses external credentials to configure OAuth details
Named Credentials
configured with named credentials for URL and other credentials
A webstore can have multiple shipping providers configured based on locales supported by the application
WebStore
Webstore Name
Locale
Shipping Provider
uses service record
HTTP Service

Configuration Details

	Section	Task	Details
1	Salesforce Org	Create a Custom Metadata Type **Shipping Provider**	This is how it looks like & further details can be found under Meta Data section : Setup > Custom Code > Custom Metadata Types > **Shipping Provider** 
2	Salesforce Org	Create a Custom Metadata Type **HTTP Service**	Here are Metadata type details - Setup > Custom Code > Custom Metadata Types > **HTTP Service**

SETUP
Custom Metadata Types

Custom Metadata Type Detail

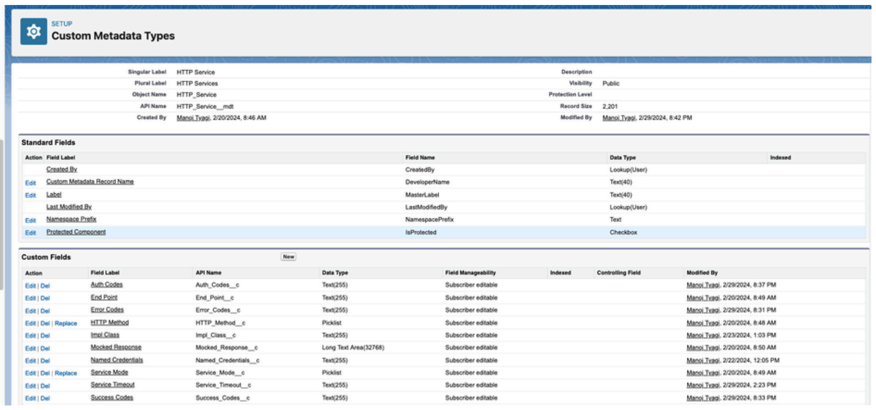
Singular Label	Shipping Provider	Description	
Plural Label	Shipping Providers	Visibility	Public
Object Name	Shipping_Provider	Protection Level	
API Name	Shipping_Provider__mdt	Record Size	1,722
Created By	Manoj Tyagi, 2/19/2024, 2:58 PM	Modified By	Manoj Tyagi, 2/29/2024, 2:20 PM

Standard Fields

Action	Field Label	Field Name	Data Type	Indexed
	Created By	CreatedBy	Lookup(User)	
Edit	Custom Metadata Record Name	DeveloperName	Text(40)	
Edit	Label	MasterLabel	Text(40)	
	Last Modified By	LastModifiedBy	Lookup(User)	
Edit	Namespace Prefix	NamespacePrefix	Text	
Edit	Protected Component	IsProtected	Checkbox	

Custom Fields

Action	Field Label	API Name	Data Type	Field Manageability	Indexed	Controlling Field	Modified By
Edit | Del	Account Number	Account_Number__c	Text(255)	Subscriber editable			Manoj Tyagi, 3/5/2024, 2:39 PM
Edit | Del	Context Locale	Context_Locale__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/29/2024, 2:12 PM
Edit | Del	Context Name	Context_Name__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/29/2024, 2:11 PM
Edit | Del	Impl Class	Impl_Class__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/21/2024, 8:37 PM
Edit | Del	Max Weight Threshold	Max_Weight_Threshold__c	Number(18, 0)	Subscriber editable			Manoj Tyagi, 2/22/2024, 10:25 AM
Edit | Del	Min Weight Threshold	Min_Weight_Threshold__c	Number(18, 0)	Subscriber editable			Manoj Tyagi, 2/22/2024, 10:25 AM
Edit | Del	Request	Request__c	Long Text Area(32768)	Subscriber editable			Manoj Tyagi, 2/20/2024, 8:31 AM
Edit | Del	Service	Service__c	Metadata Relationship(HTTP Service)	Subscriber editable	✓		Manoj Tyagi, 2/20/2024, 9:29 AM
Edit | Del	Shipping Options	Shipping_Options__c	Long Text Area(32768)	Subscriber editable			Manoj Tyagi, 2/21/2024, 8:55 PM


SETUP

Custom Metadata Types

Singular Label	HTTP Service	Description	
Plural Label	HTTP Services	Visibility	Public
Object Name	HTTP_Service	Protection Level	
API Name	HTTP_Service__mdt	Record Size	2,201
Created By	Manoj Tyagi, 2/20/2024, 8:46 AM	Modified By	Manoj Tyagi, 2/29/2024, 8:42 PM

Standard Fields

Action	Field Label	Field Name	Data Type	Indexed
	Created By	CreatedBy	Lookup(User)	
Edit	Custom Metadata Record Name	DeveloperName	Text(40)	
Edit	Label	MasterLabel	Text(40)	
	Last Modified By	LastModifiedBy	Lookup(User)	
Edit	Namespace Prefix	NamespacePrefix	Text	
Edit	Protected Component	IsProtected	Checkbox	

Custom Fields

Action	Field Label	API Name	Data Type	Field Manageability	Indexed	Controlling Field	Modified By
Edit | Del	Auth Codes	Auth_Codes__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/29/2024, 8:37 PM
Edit | Del	End Point	End_Point__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/20/2024, 8:49 AM
Edit | Del	Error Codes	Error_Codes__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/29/2024, 8:31 PM
Edit | Del | Replace	HTTP Method	HTTP_Method__c	Picklist	Subscriber editable			Manoj Tyagi, 2/20/2024, 8:48 AM
Edit | Del	Impl Class	Impl_Class__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/23/2024, 1:03 PM
Edit | Del	Mocked Response	Mocked_Response__c	Long Text Area(32768)	Subscriber editable			Manoj Tyagi, 2/20/2024, 8:50 AM
Edit | Del	Named Credentials	Named_Credentials__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/22/2024, 12:05 PM
Edit | Del | Replace	Service Mode	Service_Mode__c	Picklist	Subscriber editable			Manoj Tyagi, 2/20/2024, 8:49 AM
Edit | Del	Service Timeout	Service_Timeout__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/29/2024, 2:23 PM
Edit | Del	Success Codes	Success_Codes__c	Text(255)	Subscriber editable			Manoj Tyagi, 2/29/2024, 8:33 PM

Custom Metadata Type - Shipping Provider

Field Label	API Name	Data Type	Notes
Is Active?	Is_Active__c	Checkbox	Used to activate a shipping provider to be used
Carrier	Carrier__c	Text(255)	Stores carrier which used on UI to display shipping methods grouping by carrier
Display Order	Display_Order__c	Number(18, 0) (Unique)	Stores display order for a shipping provider, used on UI to display shipping methods when there are multiple shipping providers configured.
Account Number	Account_Number__c	Text(255)	Merchant's Shipper/Account number, it is provided by shipping provider company.
Context Locale	Context_Locale__c	Text(255)	This field stores locale , used to associate a shipping provider with a locale (one of the locales supported by a store)
Context Name	Context_Name__c	Text(255)	This field stores web store's name , used to associate a shipping provider with a web store
Impl Class	Impl_Class__c	Text(255)	Apex calls created to implement shipping provider specific details e.g. UPSShippingProvider
Max Weight Threshold	Max_Weight_Threshold__c	Number(18, 0)	Maximum weight threshold for a package
Min Weight Threshold	Min_Weight_Threshold__c	Number(18, 0)	Minimum weight threshold for a package
Request	Request__c	Long Text Area(32768)	Request JSON for a shipping provider
Service	Service__c	**Metadata Relationship(HTTP Service)**	This associate shipping provider recording with corresponding HTTP Service record
Shipping Options	Shipping_Options__c	Long Text Area(32768)	This stores shipping methods , code ↔ name mappings used to display shipping

			method names on storefront

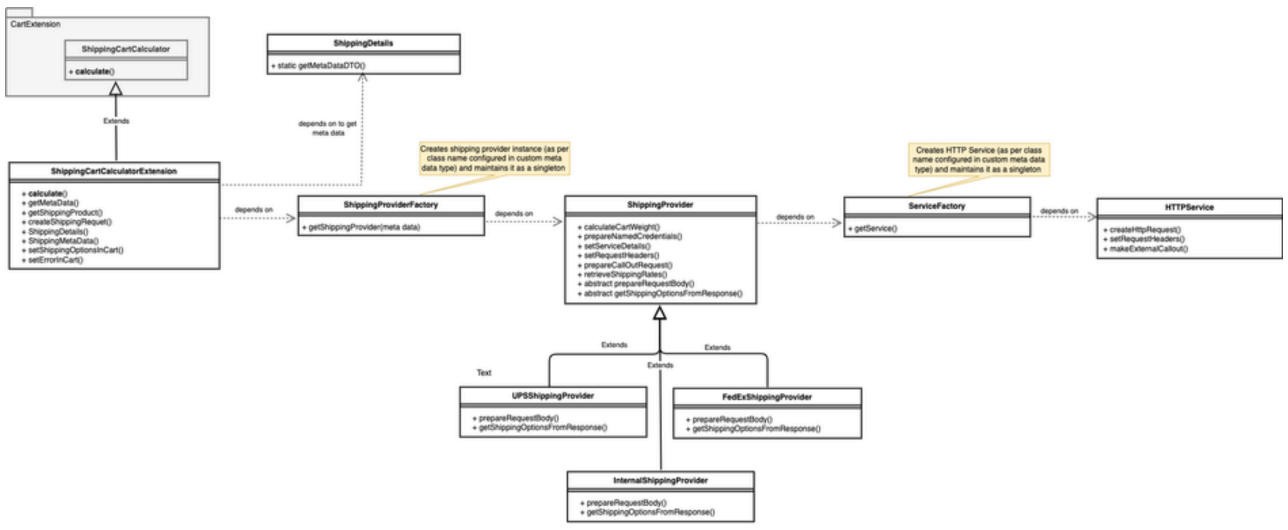
Custom Metadata Type - HTTP Service

Field Label	API Name	Data Type
Auth Codes	Auth_Codes__c	Text(255)
End Point	End_Point__c	Text(255)
Error Codes	Error_Codes__c	Text(255)
HTTP Method	HTTP_Method__c	Picklist
Impl Class	Impl_Class__c	Text(255)
Mocked Response	Mocked_Response__c	Long Text Area(32768)
Named Credentials	Named_Credentials__c	Text(255)
Service Mode	Service_Mode__c	Picklist
Service Timeout	Service_Timeout__c	Text(255)
Success Codes	Success_Codes__c	Text(255)

Product

Field Label	API Name	Data Type
Weight	Weight__c	Text(255)

Class Diagram


CartExtension
ShippingCartCalculator
+ calculate()
Extends
ShippingCartCalculatorExtension
+ calculate()
+ getMetaData()
+ getShippingProduct()
+ createShippingRequet()
+ ShippingDetails()
+ ShippingMetaData()
+ setShippingOptionsInCart()
+ setErrorInCart()
ShippingDetails
+ static getMetaDataDTO()
depends on to get meta data
depends on
Creates shipping provider instance (as per class name configured in custom meta data type) and maintains it as a singleton
ShippingProviderFactory
+ getShippingProvider(meta data)
depends on
ShippingProvider
+ calculateCartWeight()
+ prepareNamedCredentials()
+ setServiceDetails()
+ setRequestHeaders()
+ prepareCallOutRequest()
+ retrieveShippingRates()
+ abstract prepareRequestBody()
+ abstract getShippingOptionsFromResponse()
depends on
Creates HTTP Service (as per class name configured in custom meta data type) and maintains it as a singleton
ServiceFactory
+ getService()
depends on
HTTPService
+ createHttpRequest()
+ setRequestHeaders()
+ makeExternalCallout()
Extends
Extends
Extends
Text
UPSShippingProvider
+ prepareRequestBody()
+ getShippingOptionsFromResponse()
FedExShippingProvider
+ prepareRequestBody()
+ getShippingOptionsFromResponse()
InternalShippingProvider
+ prepareRequestBody()
+ getShippingOptionsFromResponse()

Source Code

Class	Details	Code
ShippingCartCalculatorExtension	This class extends **CartExtension.ShippingCartCalculator** provided by extension framework	As per need, you can extend this class & override following methods : calculate setShippingOptionsInCart setErrorInCart getMetaData getShippingProduct createShippingRequest

```
public class ShippingCartCalculatorExtension extends CartExten
  public virtual override void calculate(
    CartExtension.CartCalculateCalculatorRequest request
  ) {
    CartExtension.Cart cart = request.getCart();
    CartExtension.CartValidationOutputList cartValidationOutpu
    for (Integer i = (cartValidationOutputList.size() - 1); i
      CartExtension.CartValidationOutput cvo = cartValidationO
      if (
        cvo.getType() == CartExtension.CartValidationOutputTyp
      ) {
        cartValidationOutputList.remove(cvo);
      }
    }
    CartExtension.CartDeliveryGroupList cartDeliveryGroups = c
    if (cartDeliveryGroups.size() == 0) {
      setErrorInCart(cart, Label.No_Cart_Delivery_Group);
    } else {
      CartExtension.CartItemList cartItems = cart.getCartItems
      Integer numberOfUniqueItems = cartItems.size();
      for (Integer i = (cartDeliveryGroups.size() - 1); i >= 0
        CartExtension.CartDeliveryGroup cartDeliveryGroup = ca
          i
        );
        CartExtension.CartDeliveryGroupMethodList cartDelivery
        // Clean up the CartDeliveryGroupMethods
        for (Integer j = (cartDeliveryGroupMethods.size() - 1)
          CartExtension.CartDeliveryGroupMethod method = cartD
            j
          );
          cartDeliveryGroupMethods.remove(method);
        }
        // To clear selected Cart Delivery Group Method
        cartDeliveryGroup.setSelectedCartDeliveryGroupMethod(n

        Product2 shippingProduct = getShippingProduct();
        if (shippingProduct == null) {
          setErrorInCart(cart, Label.No_Shipping_Products_Conf
        } else {
          String shippingProductId = Id.valueOf(shippingProduc
          ShippingMetaData shippingMetaData = getMetaData(cart
          ShippingProviderRequest shippingRequest = createShip
            cartDeliveryGroup
          );
          shippingRequest.shippingMetaData = shippingMetaData;
          ShippingProvider sProvider = ShippingProviderFactory
            shippingMetaData
          );
          if (sProvider != null) {
            shippingRequest.cartId = cart.getId();
            Map<String, ShippingProviderResponse> shippingMeth
              shippingRequest
            );
            if (
              shippingMethodsWithRate != null &&
              shippingMethodsWithRate.size() > 0
            ) {
              setShippingOptionsInCart(
                shippingMethodsWithRate,
                cartDeliveryGroupMethods,
                shippingProductId,
                shippingMetaData
              );
            } else {
              setErrorInCart(cart, Label.Failed_to_get_shippin
            }
          }
        }
      }
    }
  }

  private virtual void setShippingOptionsInCart(
    Map<String, ShippingProviderResponse> shippingMethodsWithR
    CartExtension.CartDeliveryGroupMethodList cartDeliveryGrou
    String shippingProduct,
    ShippingMetaData shippingMetaData
  ) {
    try {
      Map<String, String> shippingMethodNames = (Map<String, S
        shippingMetaData.shippinggMethodNames,
        Map<String, String>.class
      );
      for (String serviceCode : shippingMethodsWithRate.keySet
        CartExtension.CartDeliveryGroupMethod cartDeliveryGrou
          shippingMethodNames.get(serviceCode),
          shippingMethodsWithRate.get(serviceCode).cost,
          shippingProduct
        );
        cartDeliveryGroupMethod.setExternalProvider(serviceCod
        cartDeliveryGroupMethod.setClassOfService(
          shippingMethodNames.get(serviceCode)
        );
        cartDeliveryGroupMethod.setIsActive(true);
        cartDeliveryGroupMethods.add(cartDeliveryGroupMethod);
      }
    } catch (Exception expObj) {
      System.debug(
        'Exception due to error ====' +
          expObj.getMessage() +
          'at Line Number ====' +
          expObj.getLineNumber()
      );
      // WK_Exception.log(expObj, applicationName, moduleName,
      // methodName, supportData);
    }
  }
  private virtual void setErrorInCart(
    CartExtension.Cart cart,
    String errorMessage
  ) {
    CartExtension.CartValidationOutput cvo = new CartExtension
      CartExtension.CartValidationOutputTypeEnum.SHIPPING,
      CartExtension.CartValidationOutputLevelEnum.ERROR
    );
    cvo.setMessage(errorMessage);
    CartExtension.CartValidationOutputList cartValidationOutpu
    cartValidationOutputList.add(cvo);
  }

  public virtual ShippingMetaData getMetaData(String webStoreI
    return ShippingDetails.getMetaDataDTO(webStoreId);
  }

  public virtual Product2 getShippingProduct() {
    Product2 shippingProduct;
    List<Product2> shippingProducts = [
      SELECT Id
      FROM Product2
      WHERE product2.Family = 'Shipping'
      LIMIT 1
    ];
    if (shippingProducts.size() > 0) {
      shippingProduct = shippingProducts[0];
    }
    return shippingProduct;
  }

  public virtual ShippingProviderRequest createShippingRequest
    CartExtension.CartDeliveryGroup cartDeliveryGroup
  ) {
    ShippingProviderRequest request = new ShippingProviderRequ
    request.street = cartDeliveryGroup.getDeliverToAddress().S
    request.city = cartDeliveryGroup.getDeliverToAddress().Cit
    request.state = cartDeliveryGroup.getDeliverToAddress().St
    request.postalCode = cartDeliveryGroup.getDeliverToAddress
    request.country = cartDeliveryGroup.getDeliverToAddress().

    return request;
  }
}

```

ShippingDetails	This class fetches custom meta data and stores in a map so that can be used further down in the execution flow	(code below)

```
public with sharing class ShippingDetails {
  private static Map<String, ShippingMetaData> shippingProvider
  private static Shipping_Provider__mdt shippingProviderMDT;
  private ShippingDetails() {
  }

  public static ShippingMetaData getMetaDataDTO(String contextI
    ShippingMetaData shippingMetaDataDTO;
    String currentLocale = UserInfo.getLocale();
    String dataKey = contextId + '-' + currentLocale;
    if (shippingProviders.containsKey(dataKey)) {
      shippingMetaDataDTO = shippingProviders.get(dataKey);
    } else {
      String webStoreName = [
        SELECT Name
        FROM WebStore
        WHERE Id = :contextId
        LIMIT 1
      ]
      .Name;
      shippingProviderMDT = [
        SELECT
          Id,
          Impl_Class__c,
          Max_Weight_Threshold__c,
          Min_Weight_Threshold__c,
          QualifiedApiName,
          Request__c,
          Account_Number__c,
          Service__c,
          Shipping_Options__c,
          Service__r.End_Point__c,
          Service__r.HTTP_Method__c,
          Service__r.Mocked_Response__c,
          Service__r.Service_Mode__c,
          Service__r.Service_Timeout__c,
          Service__r.Impl_Class__c,
          Service__r.Named_Credentials__c
        FROM Shipping_Provider__mdt
        WHERE
          Context_Name__c = :webStoreName
          AND Context_Locale__c = :currentLocale
        LIMIT 1
      ];
      if (shippingProviderMDT != null) {
        shippingMetaDataDTO = new ShippingMetaData();
        shippingMetaDataDTO.requestJSON = shippingProviderMDT.R
        shippingMetaDataDTO.provideImplClass = shippingProvider
        shippingMetaDataDTO.maxPackageWeight = shippingProvider
        shippingMetaDataDTO.minPackageWeight = shippingProvider
        shippingMetaDataDTO.shippinggMethodNames = shippingProv
        shippingMetaDataDTO.accountNumber = shippingProviderMDT
        shippingMetaDataDTO.endPoint = shippingProviderMDT.Serv
        shippingMetaDataDTO.httpMethod = shippingProviderMDT.Se
        shippingMetaDataDTO.mockedResponse = shippingProviderMD
        shippingMetaDataDTO.serviceMode = shippingProviderMDT.S
        shippingMetaDataDTO.serviceTimeout = shippingProviderMD
        shippingMetaDataDTO.serviceImplClass = shippingProvider
        shippingMetaDataDTO.namedCredentials = shippingProvider
        shippingProviders.put(dataKey, shippingMetaDataDTO);
      }
    }
    return shippingMetaDataDTO;
  }

  public static Shipping_Provider__mdt getMetaDataObject() {
    return shippingProviderMDT;
  }
}
```

ShippingProviderFactory	This class works as a factory to create shipping provider singletons as per **Impl Class** configured in custom meta data record	`public with sharing class ShippingProviderFactory {`
		`  private static ShippingProvider shippingProvider;`
		`  private ShippingProviderFactory() {`
		`  }`
		`  public static ShippingProvider getShippingProvider(`
		`    ShippingMetaData shippingMetaData`
		`  ) {`
		`    if (shippingProvider == null) {`
		`      if (shippingMetaData.provideImplClass != null) {`
		`        Type t = Type.forName(shippingMetaData.provideImplClass`
		`        shippingProvider = (ShippingProvider) t.newInstance();`
		`      }`
		`    }`
		`    return shippingProvider;`
		`  }`
		`}`

ShippingProvider	This class provides common functionality to prepare a shipping request and should be further extended by a specific shipping providers like UPS , FedEx.	You must extend this class & override following abstract methods
		
		prepareRequestBody
		getShippingOptionsFromResponse
		
		As per need, overriding following methods is optional
		
		calculateCartWeight
		prepareNamedCredentials
		setServiceDetails
		setRequestHeaders
		prepareCallOutRequest
		retrieveShippingRates
		
		`public abstract class ShippingProvider {`
		`  // Calculate Product Weight here -`
		`  // The maximum per package weight for the selected service f`
		`  // country or territory is 150.00 pounds.`
		`  public virtual List<Decimal> calculateCartWeight(`
		`    ShippingProviderRequest shippingRequest`
		`  ) {`
		`    Decimal shippingMaxWeight = 150; // weight in lbs`
		`    Decimal shippingMinWeight = 5; // weight in lbs`
		`    Decimal productWeight = 0;`
		`    String cartId = shippingRequest.cartId;`
		`    if (shippingRequest.shippingMetaData.maxPackageWeight != n`
		`      shippingMaxWeight = shippingRequest.shippingMetaData.max`
		`    }`
		`    if (shippingRequest.shippingMetaData.minPackageWeight != n`
		`      shippingMinWeight = shippingRequest.shippingMetaData.min`
		`    }`
		
		`    List<CartItem> lstCartItems = [`
		`      SELECT`
		`        Product2Id,`
		`        Product2.weight__c,`
		`        Name,`
		`        Id,`

```
      CartId,
      Type,
      Sku,
      Quantity,
      ListPrice,
      SalesPrice,
      TotalListPrice
    FROM CartItem
    WHERE CartId = :cartId AND Type = 'Product'
  ];

  for (CartItem cartItem : lstCartItems) {
    productWeight +=
      cartItem.Quantity * Decimal.ValueOf(cartItem.Product2.
  }
  List<Decimal> lstShippingWeight = new List<Decimal>();
  while (productWeight > shippingMaxWeight) {
    productWeight = productWeight - shippingMaxWeight;
    lstShippingWeight.add(shippingMaxWeight);
  }
  if (productWeight < shippingMinWeight) {
    productWeight = 5;
  }
  lstShippingWeight.add(productWeight);
  return lstShippingWeight;
}

public virtual Map<String, String> prepareNamedCredentials(
  ShippingProviderRequest shippingRequest
) {
  Map<String, String> callOutRequest = new Map<String, Strin
  String endPoint =
    Constants.CALLOUT +
    shippingRequest.shippingMetaData.namedCredentials +
    shippingRequest.shippingMetaData.endPoint;
  callOutRequest.put(Constants.END_POINT, endPoint);
  // callOutRequest.put(Constants.USERNAME , Constants.CREDE
  // callOutRequest.put(Constants.PASSWORD , Constants.CREDE
  return callOutRequest;
}

public virtual Map<String, String> setServiceDetails(
  ShippingProviderRequest shippingRequest
) {
  Map<String, String> serviceDetails = new Map<String, Strin
  serviceDetails.put(
    Constants.HTTP_METHOD,
    shippingRequest.shippingMetaData.httpMethod
  );
  serviceDetails.put(
    Constants.SERVICE_TIMEOUT,
    shippingRequest.shippingMetaData.serviceTimeout
  );
  serviceDetails.put(
    Constants.SERVICE_MODE,
    shippingRequest.shippingMetaData.serviceMode
  );
  serviceDetails.put(
    Constants.MOCKED_RESPONSE,
    shippingRequest.shippingMetaData.mockedResponse
  );
  return serviceDetails;
}

public virtual Map<String, String> setRequestHeaders(
  ShippingProviderRequest shippingRequest
) {
  Map<String, String> htttpRequestDetails = new Map<String,
  return htttpRequestDetails;
}
public virtual Map<String, String> prepareCallOutRequest(
  ShippingProviderRequest shippingRequest
) {
  Map<String, String> callOutRequest = new Map<String, Strin
  callOutRequest.putAll(prepareNamedCredentials(shippingRequ
  callOutRequest.putAll(setServiceDetails(shippingRequest));
  prepareRequestBody(shippingRequest, callOutRequest);
  return callOutRequest;
}

public virtual Map<String, ShippingProviderResponse> retriev
  ShippingProviderRequest shippingRequest
) {
  List<String> responseList = new List<String>();
  Map<String, ShippingProviderResponse> shippingMethodsWithR
  Map<String, String> responseMap;
  Boolean calloutSuccess = true;
  try {
    List<Decimal> lstShippingWeight = calculateCartWeight(sh
    for (Integer i = (lstShippingWeight.size() - 1); i >= 0;
      shippingRequest.packageWeight = lstShippingWeight.get(

      Map<String, String> callOutRequest = prepareCallOutReq
        shippingRequest
      );
      responseMap = ServiceFactory.getService(
          shippingRequest.shippingMetaData.serviceImplClass
        )
        .makeExternalCallout(
          callOutRequest,
          setRequestHeaders(shippingRequest)
        );
      if (
        !'200'.equals(responseMap.get(Constants.RESPONSE_REA
        responseMap.isEmpty()
      ) {
        calloutSuccess = false;
        break;
      }
      responseList.add(responseMap.get(Constants.SERVICE_RES
    }
    if (calloutSuccess) {
      shippingMethodsWithRate = getShippingOptionsFromRespon
        responseList,
        shippingRequest
      );
    }
  } catch (Exception expObj) {
    System.debug(
      'Exception due to error ====' +
        expObj.getMessage() +
        'at Line Number ====' +
        expObj.getLineNumber()
    );
    // WK_Exception.log(expObj, applicationName, moduleName,
    // methodName, supportData);
  }

  return shippingMethodsWithRate;
}
public abstract Map<String, ShippingProviderResponse> getShi
  List<String> responseList,
  ShippingProviderRequest shippingRequest
);
public abstract void prepareRequestBody(
  ShippingProviderRequest shippingRequest,
  Map<String, String> callOutRequest
);
}
```

ShippingProviderRequest

This class is used as a DTO to transfer request specific data & meta data between classes

```
public with sharing class ShippingProviderRequest {
  public ShippingProviderRequest() {
  }

  public String street { get; set; }
  public String city { get; set; }
  public String state { get; set; }
  public String postalCode { get; set; }
  public String country { get; set; }
  public String cartId { get; set; }
  public Decimal packageWeight { get; set; }
  public ShippingMetaData shippingMetaData { get; set; }

  private Map<String, Object> additionalData = new Map<String,

  public Object getData(String key) {
    return additionalData.get(key);
  }

  public void addData(String key, Object value) {
    additionalData.put(key, value);
  }
}
```

ShippingProviderResponse

This class used as a DTO to transfer shipping rates back to shipping extension class

```
public with sharing class ShippingProviderResponse {
  public ShippingProviderResponse() {
  }


  public String serviceCode { get; set; }
  public Decimal cost { get; set; }
  public String shipDate { get; set; }
  public String shipTime { get; set; }

  private Map<String, Object> additionalData = new Map<String,

  public Object getData(String key) {
    return additionalData.get(key);
  }

  public void addData(String key, Object value) {
    additionalData.put(key, value);
  }

}

```

ServiceFactory

This class works as a factory to create HTTP Service singletons as per **Impl Class** configured in custom meta data record for service. For now there is no shipping specific class so it returns base class itself i.e. HTTPService

```
public with sharing class ServiceFactory {
  private static HTTPService service;
  private ServiceFactory() {
  }
  public static HTTPService getService(String className) {
    if (service == null) {
      if (String.isNotEmpty(className)) {
        Type t = Type.forName(className);
        service = (HTTPService) t.newInstance();
      } else {
        service = new HTTPService();
      }
    }
    return service;
  }
}

```

HTTPService

This class is used to make a REST based callout. Currently supporting only JSON as data format.

As per need, you can extend this class and override following methods :

createHttpRequest
setRequestHeaders
makeExternalCallout

```
public virtual class HTTPService {
  public HTTPService() {
  }
  public virtual HttpRequest createHttpRequest(
    Map<String, String> requestDetails
  ) {
    HttpRequest req = new HttpRequest();
    req.setEndpoint(requestDetails.get(Constants.END_POINT));
    req.setMethod(
      String.isNotBlank(requestDetails.get(Constants.HTTP_METH
        ? requestDetails.get(Constants.HTTP_METHOD)
        : Constants.HTTP_POST
    );
    Integer timeout = String.isNotBlank(
        requestDetails.get(Constants.SERVICE_TIMEOUT)
      )
      ? Integer.valueOf(requestDetails.get(Constants.SERVICE_T
      : Constants.HTTP_DEFAULT_TIMEOUT;
    req.setTimeout(timeout);
    return req;
  }

  public virtual void setRequestHeaders(
    HttpRequest req,
    Map<String, String> requestHeaders
  ) {
    if (!requestHeaders.isEmpty()) {
      for (String key : requestHeaders.keySet()) {
        req.setHeader(key, requestHeaders.get(key));
      }
    }
    if (!requestHeaders.containsKey(Constants.HTTP_HEADER_CONT
      req.setHeader(
        Constants.HTTP_HEADER_CONTENT_TYPE,
        Constants.HTTP_HEADER_CONTENT_TYPE_JSON
      );
    }
  }

  public virtual Map<String, String> makeExternalCallout(
    Map<String, String> calloutRequestDetails,
    Map<String, String> requestHeaders
  ) {
    Map<String, String> responseMap = new Map<String, String>(
    try {
      HttpRequest req = createHttpRequest(calloutRequestDetail
      setRequestHeaders(req, requestHeaders);
      Http http = new Http();
      HTTPResponse res = null;
      if (!Test.isRunningTest()) {
        if (
          calloutRequestDetails.get(Constants.SERVICE_MODE)
            .toUpperCase()
            .equals(Constants.SERVICE_MODE_LIVE)
        ) {
          req.setbody(
            calloutRequestDetails.get(Constants.SERVICE_REQUES
          );
          res = http.send(req);
        } else {
          res = new HTTPResponse();
          res.setStatusCode(Constants.HTTP_200);
          res.setBody(calloutRequestDetails.get(Constants.MOCK
        }
      } else {
        res = new HTTPResponse();
        res.setStatusCode(Constants.HTTP_200);
      }
      if (res.getStatusCode() == Constants.HTTP_200) {
        responseMap.put(
          Constants.HTTP_RESPONSE_STATUS,
          Constants.HTTP_RESPONSE_STATUS_SUCCESS
        );
        responseMap.put(
          Constants.RESPONSE_REASON_CODE,
          String.valueOf(res.getstatusCode())
        );
        responseMap.put(Constants.SERVICE_RESPONSE_BODY, res.g
      } else {
        responseMap.put(
          Constants.HTTP_RESPONSE_STATUS,
          Constants.HTTP_RESPONSE_STATUS_ERROR
        );
        responseMap.put(
          Constants.RESPONSE_REASON_CODE,
          String.valueOf(res.getstatusCode())
        );
        responseMap.put(Constants.SERVICE_RESPONSE_BODY, res.g
      }
    } catch (Exception expObj) {
      responseMap.put(
        Constants.HTTP_RESPONSE_STATUS,
        Constants.HTTP_RESPONSE_STATUS_ERROR
      );
      System.debug(
        'Exception due to error ====' +
          expObj.getMessage() +
          'at Line Number ====' +
          expObj.getLineNumber()
      );
      // WK_Exception.log(expObj, applicationName, moduleName,
      // methodName, supportData);
    }
    return responseMap;
  }

  public virtual void setAuth() {
  }
  public virtual void setAuthHeaders() {
  }
}

```