Oracle® Hospitality

OHEICS UI Support Services

Release 18.1

Part Number E90857-02

December 2020

Contents

Tables

Preface

This document contains API integration information for Oracle Hospitality eCommerce Integration Cloud Service.

Audience

This document is intended integrators to Oracle Hospitality eCommerce Integration Cloud Service.

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received and any associated log files
- Screenshots of each step you take

Documentation

Oracle Hospitality product documentation is available on the Oracle Help Center at http://docs.oracle.com/en/industries/hospitality/

Revision History

Date	Description of Change
March 2018	• Initial publication
December 2020	• Updated Sample Code (C#) in Chapter 1: Services Overview

1 Services Overview

The OHEICS UI Support Services are designed to facilitate web clients in the process of placing an order. There are currently six services:

- Security Service – responsible for initiating a client session with the system.
- Store Service – responsible for store location.
- Order Service – responsible for all ordering functionality from getting a stores menu, to basket operations and order placement.
- Payment Service – responsible for processing payment for orders.
- Customer Service – responsible for customer centric functionality such as login/logout and customer account management.
- Check Service – responsible for check interactions with POS.

Connection Protocols

The OHEICS web services use TLS 1.2 to exchange structured information over https.

Authentication

Authentication to the OHEICS web services is performed by including some information in each request. This is then validated by the web services before access is granted. The two values need to be generated by the client are a TimeStampToken and a Signature.

Once a TimeStampToken is generated it is used in the creation of the Signature value.

Time Stamp Token

TimeStampToken is a specially-formatted string of the current date and time.

TimeStampToken must be the current date and time in UTC.

It must be in the following, ISO-8601 compatible, format:

yyyy-**MM**-**dd**T**HH**:**mm**:**ss**.**fffffff**Z

Item	Length	Value	Valid examples
yyyy	4 digits	Year	2017
MM	2 digits	Month	02, 12

dd	2 digits	Day	01, 28
HH	2 digits	Hours (24 clock)	00, 06, 23
mm	2 digits	Minutes	00, 08, 59
ss	2 digits	Seconds	00, 02, 59
fffffff	7 digits	Fractions of a second	0000000, 5478234

For example, the UTC date & time May 21 2017 1:58:03 PM would be written as "2017-05-21T13:58:03.0000000Z"

Times **must** be given in UTC and **must** conform exactly to the format specified above. This includes the full seven digits for fractional seconds.

The following examples are invalid:

Invalid string	Reason
2017-15-21T13:37:03.0000000Z	Invalid Month – 15
2017-05-21T13:37:03.0000000	Invalid time zone - missing "Z" at end
2017-05-21T13:37:03.000000Z	Invalid number of fractional seconds - must be 7 digits

This format is refered to as "Round Trip". In the .NET Framework this format is supported by the DateTime format string "o" and the DateTimeStyles.RoundTripKind style.

Signature

This is a string generated using a TimeStampToken along with the Service and Method Names being called. These values are then cryptographically hashed using the Shared Secret Key to produce the Signature value.

The procedure for generating the Signature value is as follows:

- Concatenate ServiceName, MethodName and TimeStampToken strings
- Convert this string to a byte array using ASCII encoding
- Hash the byte array with HMAC SHA-1 using Shared Secret Key as the key
- Convert the hash result from a byte array to an upper-case hexadecimal string

Sample Code (C#)

This code is not intended for production use, but merely to help illustrate the procedure to follow when generating a Signature.

```
public string CreateSignature(string sharedSecret, string serviceName, string
methodName, string timeStampToken)
{
  //use ASCII encoding for converting strings to bytes and back
  var asciiEncoding = new System.Text.ASCIIEncoding();

  //concatenate the required strings
  string combinedString = serviceName + methodName + timeStampToken;

  //convert the concatenated string to a byte array
  byte[] combinedBytes = asciiEncoding.GetBytes(combinedString);

  //convert sharedSecret to a byte array
  byte[] sharedSecretBytes = Convert.FromBase64String(sharedSecret);

  //create cryptographic hasher for HMAC-SHA1 using shared secret
  var hmacsha1 = new System.Security.Cryptography.HMACSHA1(sharedSecretBytes);

  //compute the hash
  byte[] hashedBytes = hmacsha1.ComputeHash(combinedBytes);

  //convert byte array to a single string using "X2" format,
  //uppercase 2-digit hexadecimal. e.g. 255 -> FF, 0 -> 00, 11 -> 0B
  IEnumerable<string> hashedStrings = hashedBytes.Select(a => a.ToString("X2"));
  string signature = string.Concat(hashedStrings);

  return signature;
}
```

Using Time Stamp Token and Signature in requests

TimeStampToken and Signature are usually set in the SessionDTO object.

When there is no session, such as when requesting a new session, the main request object contains fields for TimeStampToken and Signature.

IMPORTANT: In any request the value of TimeStampToken **must** be the same as the one used to generate the value of Signature. If the two are not the same then the Signature will not be accepted by the web services.

2 Workflow

This section describes the UI Support Services workflow.

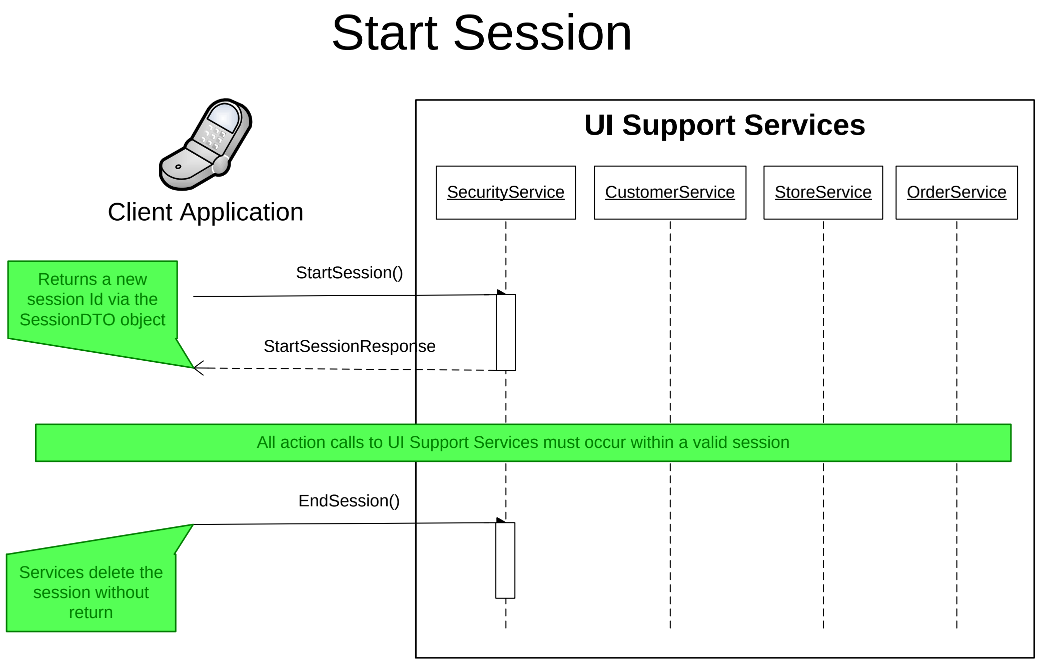
State Management

Unlike many web services, the UI Support services are not stateless. They require a calling application to call the web methods in a pre-defined order to progress to the point where a customer order can be placed.

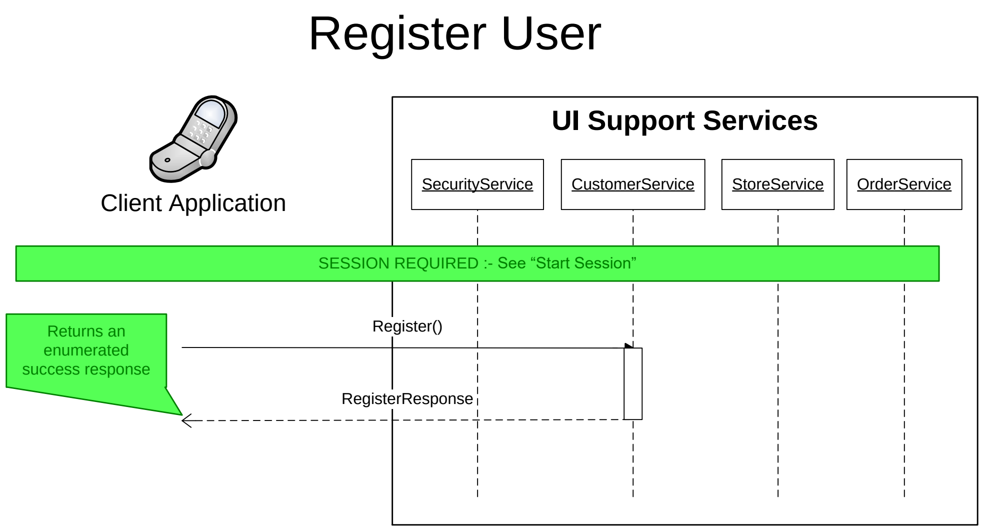
Common Use Cases

The following section describes common use cases for clients using the OHEICS UI Support Services.

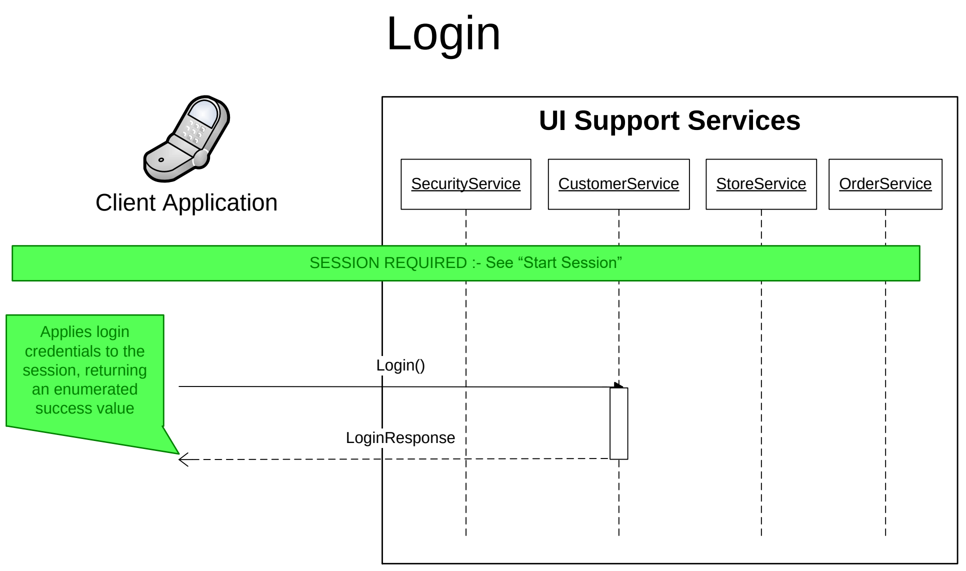
Start Session


Start Session
UI Support Services
Client Application
SecurityService
CustomerService
StoreService
OrderService
StartSession()
StartSessionResponse
Returns a new session Id via the SessionDTO object
All action calls to UI Support Services must occur within a valid session
EndSession()
Services delete the session without return

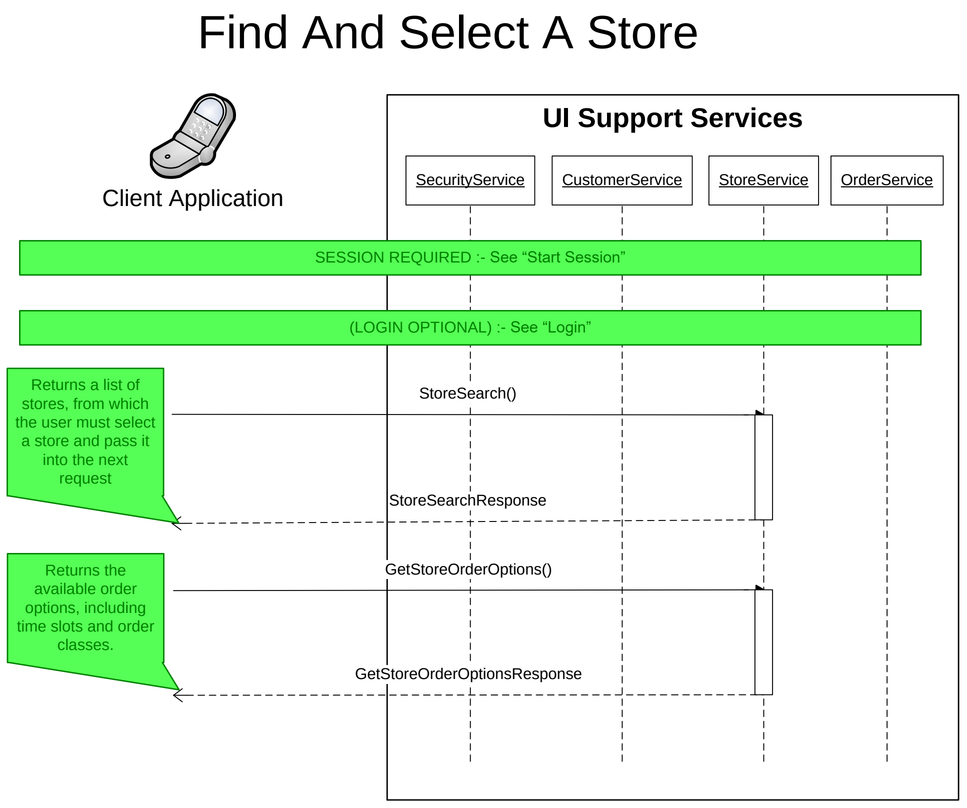
Register User



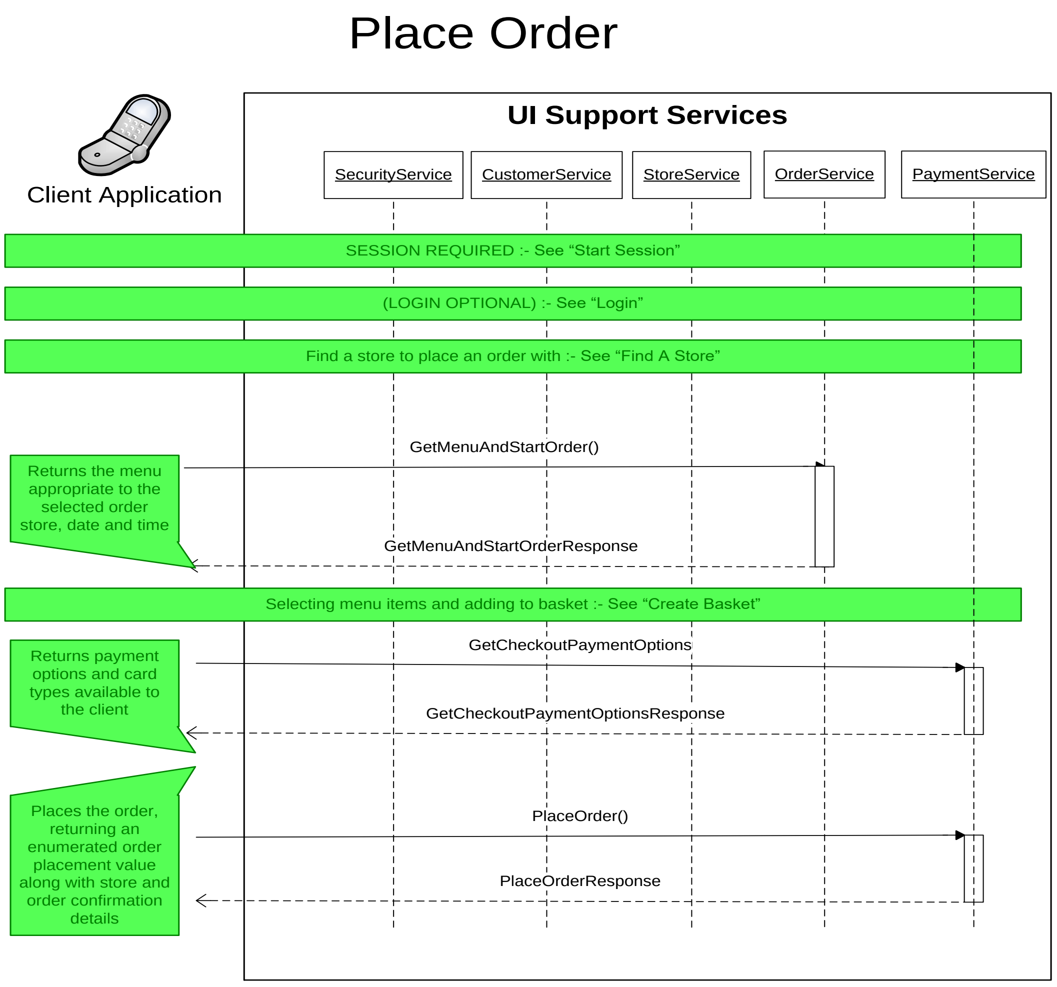
Login


Login
UI Support Services
Client Application
SecurityService
CustomerService
StoreService
OrderService
SESSION REQUIRED :- See “Start Session”
Applies login credentials to the session, returning an enumerated success value
Login()
LoginResponse

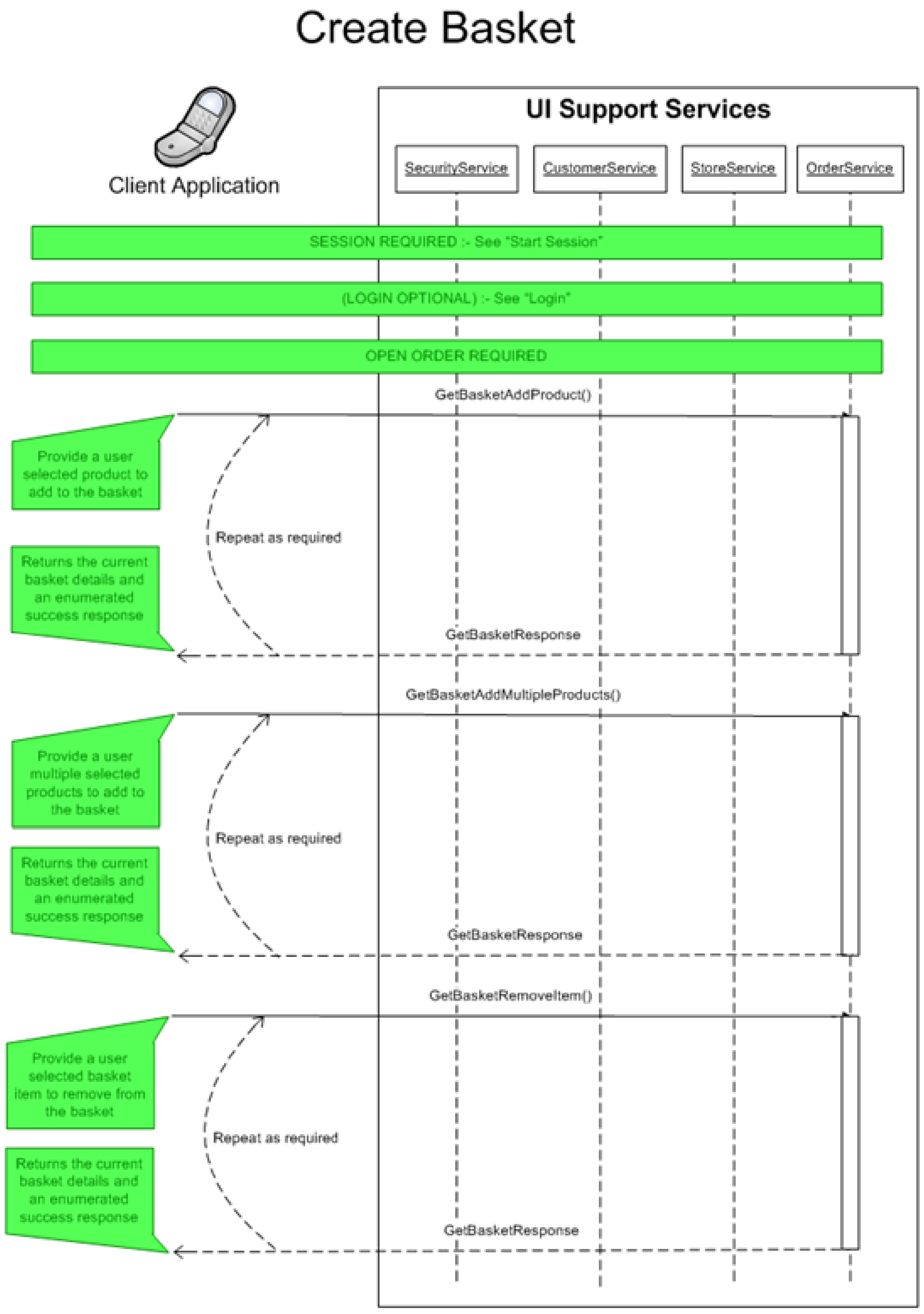
Find a Store


Find And Select A Store
Client Application
UI Support Services
SecurityService
CustomerService
StoreService
OrderService
SESSION REQUIRED :- See “Start Session”
(LOGIN OPTIONAL) :- See “Login”
Returns a list of stores, from which the user must select a store and pass it into the next request
StoreSearch()
StoreSearchResponse
Returns the available order options, including time slots and order classes.
GetStoreOrderOptions()
GetStoreOrderOptionsResponse

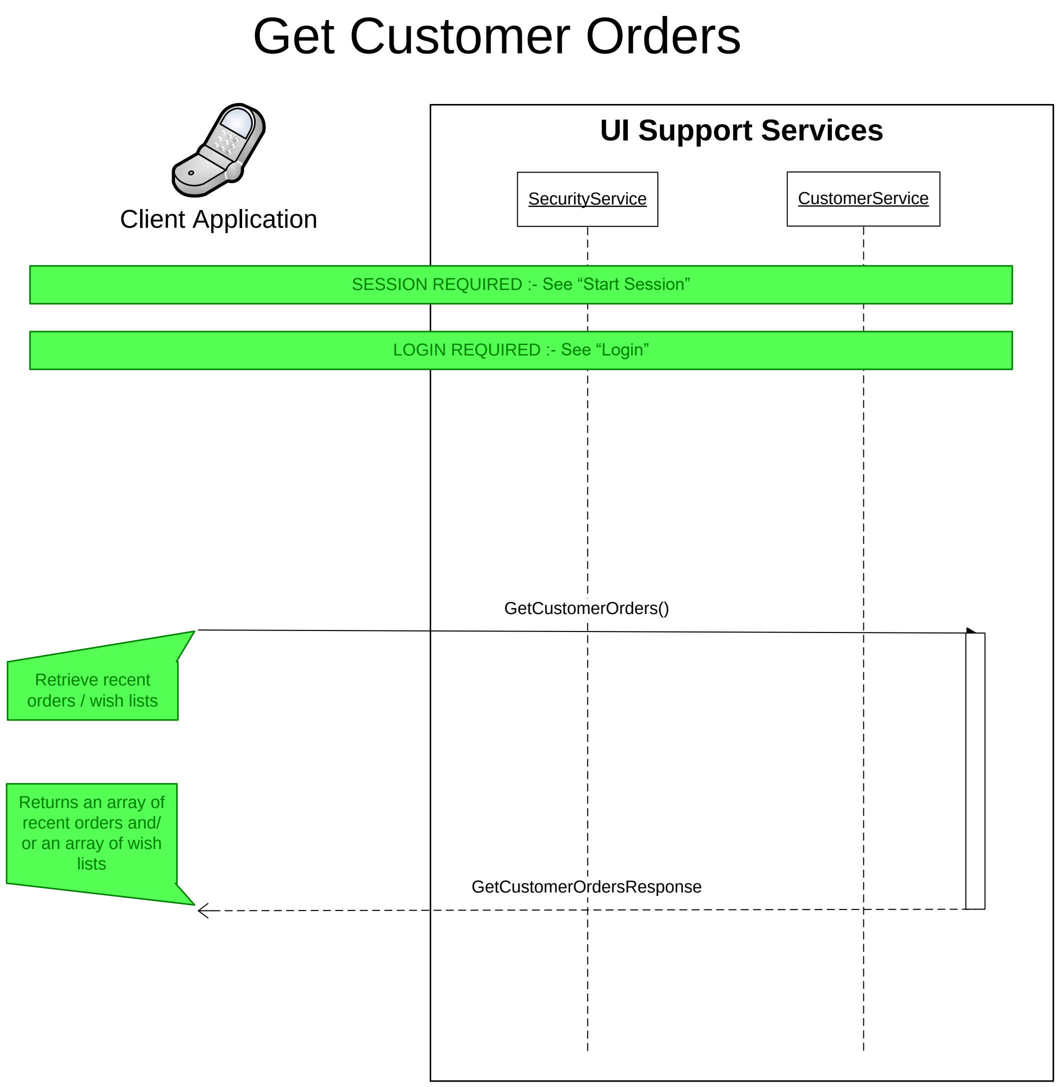
Place Order


Place Order
Client Application
UI Support Services
SecurityService
CustomerService
StoreService
OrderService
PaymentService
SESSION REQUIRED :- See "Start Session"
(LOGIN OPTIONAL) :- See "Login"
Find a store to place an order with :- See "Find A Store"
GetMenuAndStartOrder()
Returns the menu appropriate to the selected order store, date and time
GetMenuAndStartOrderResponse
Selecting menu items and adding to basket :- See "Create Basket"
GetCheckoutPaymentOptions
Returns payment options and card types available to the client
GetCheckoutPaymentOptionsResponse
Places the order, returning an enumerated order placement value along with store and order confirmation details
PlaceOrder()
PlaceOrderResponse

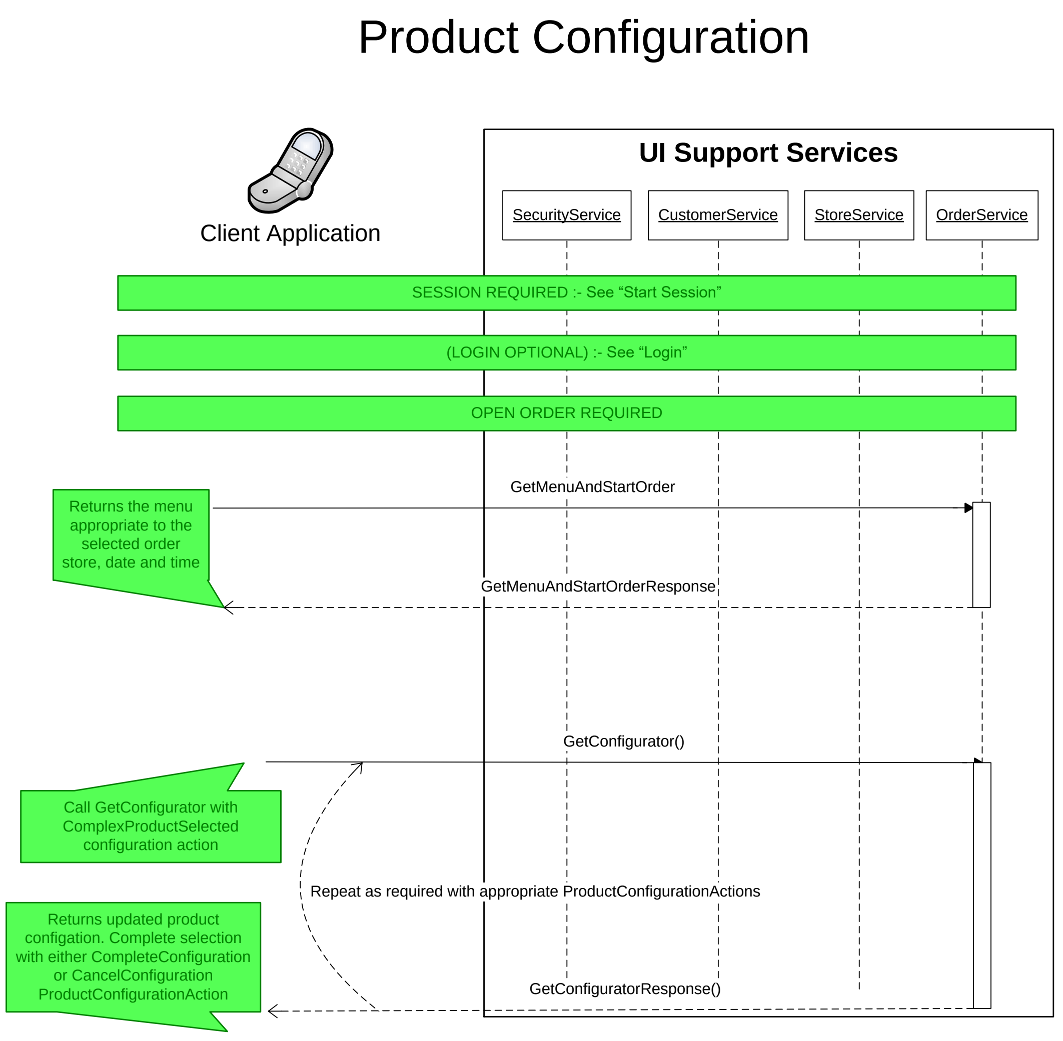
Create Basket

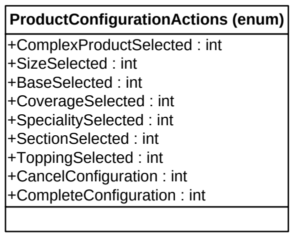

Create Basket
Client Application
UI Support Services
SecurityService
CustomerService
StoreService
OrderService
SESSION REQUIRED :- See "Start Session"
(LOGIN OPTIONAL) :- See "Login"
OPEN ORDER REQUIRED
GetBasketAddProduct()
Provide a user selected product to add to the basket
Repeat as required
Returns the current basket details and an enumerated success response
GetBasketResponse
GetBasketAddMultipleProducts()
Provide a user multiple selected products to add to the basket
Repeat as required
Returns the current basket details and an enumerated success response
GetBasketResponse
GetBasketRemoveItem()
Provide a user selected basket item to remove from the basket
Repeat as required
Returns the current basket details and an enumerated success response
GetBasketResponse

Get Customer Orders


Get Customer Orders
Client Application
UI Support Services
SecurityService
CustomerService
SESSION REQUIRED :- See “Start Session”
LOGIN REQUIRED :- See “Login”
GetCustomerOrders()
Retrieve recent orders / wish lists
Returns an array of recent orders and/ or an array of wish lists
GetCustomerOrdersResponse

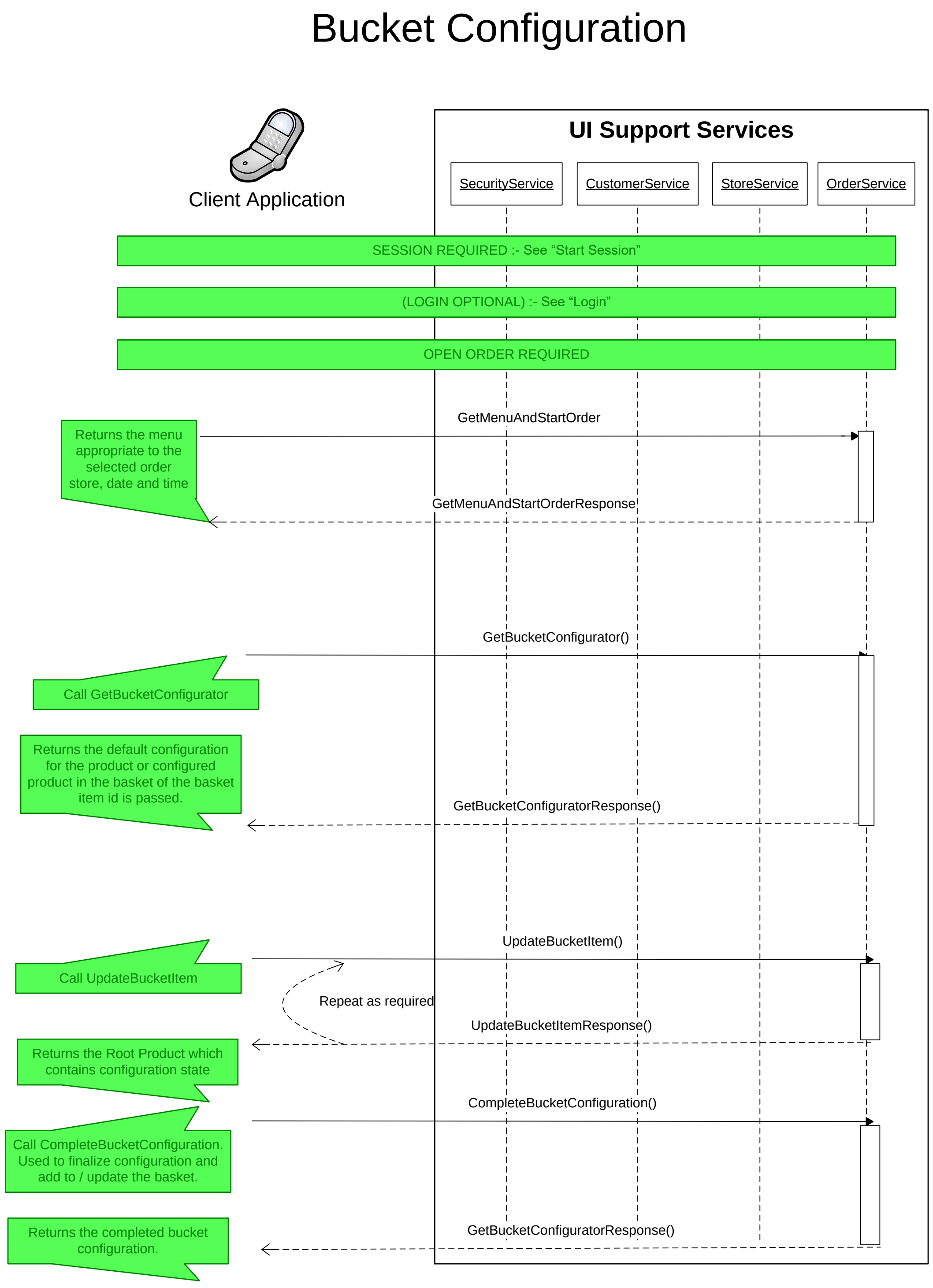
Product Configuration


Product Configuration
Client Application
UI Support Services
SecurityService
CustomerService
StoreService
OrderService
SESSION REQUIRED :- See “Start Session”
(LOGIN OPTIONAL) :- See “Login”
OPEN ORDER REQUIRED
GetMenuAndStartOrder
Returns the menu appropriate to the selected order store, date and time
GetMenuAndStartOrderResponse
GetConfigurator()
Call GetConfigurator with ComplexProductSelected configuration action
Repeat as required with appropriate ProductConfigurationActions
Returns updated product configation. Complete selection with either CompleteConfiguration or CancelConfiguration ProductConfigurationAction
GetConfiguratorResponse()

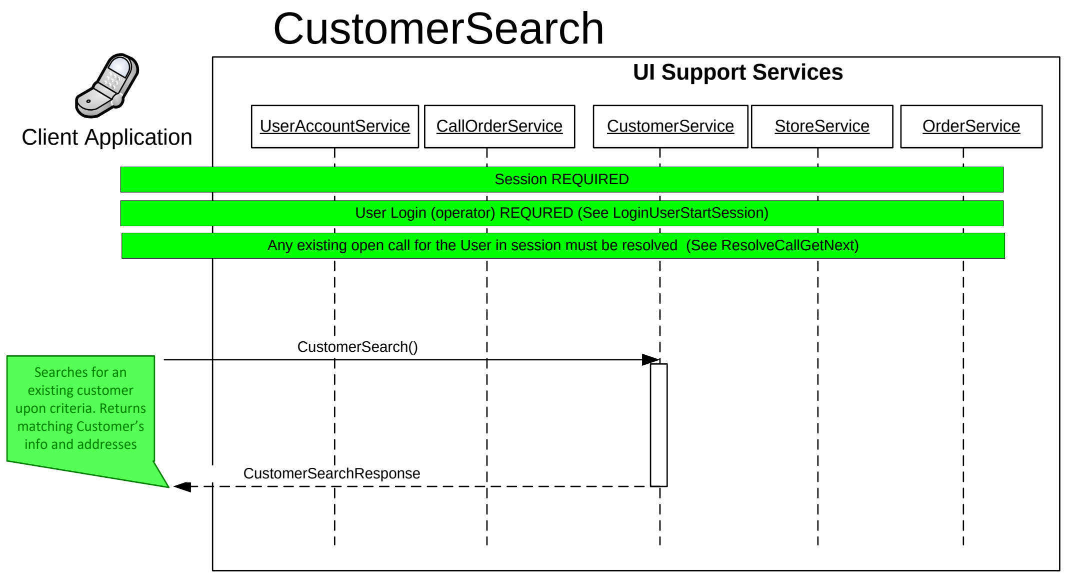

ProductConfigurationActions (enum)

+ComplexProductSelected : int
+SizeSelected : int
+BaseSelected : int
+CoverageSelected : int
+SpecialitySelected : int
+SectionSelected : int
+ToppingSelected : int
+CancelConfiguration : int
+CompleteConfiguration : int

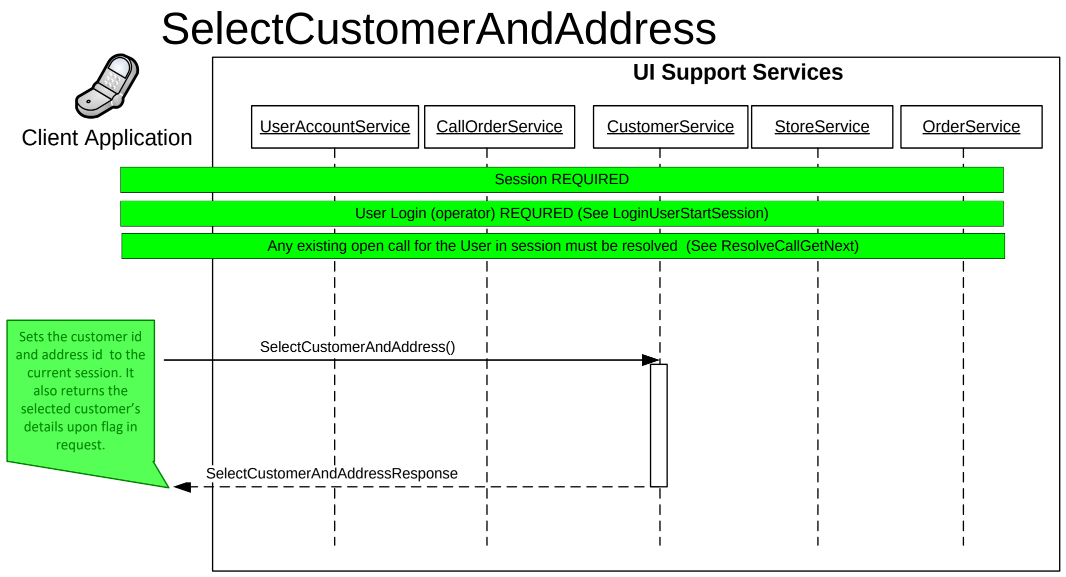
Bucket Configuration


Bucket Configuration
Client Application
UI Support Services
SecurityService
CustomerService
StoreService
OrderService
SESSION REQUIRED :- See “Start Session”
(LOGIN OPTIONAL) :- See “Login”
OPEN ORDER REQUIRED
GetMenuAndStartOrder
Returns the menu appropriate to the selected order store, date and time
GetMenuAndStartOrderResponse
GetBucketConfigurator()
Call GetBucketConfigurator
Returns the default configuration for the product or configured product in the basket of the basket item id is passed.
GetBucketConfiguratorResponse()
UpdateBucketItem()
Call UpdateBucketItem
Repeat as required
UpdateBucketItemResponse()
Returns the Root Product which contains configuration state
CompleteBucketConfiguration()
Call CompleteBucketConfiguration. Used to finalize configuration and add to / update the basket.
GetBucketConfiguratorResponse()
Returns the completed bucket configuration.

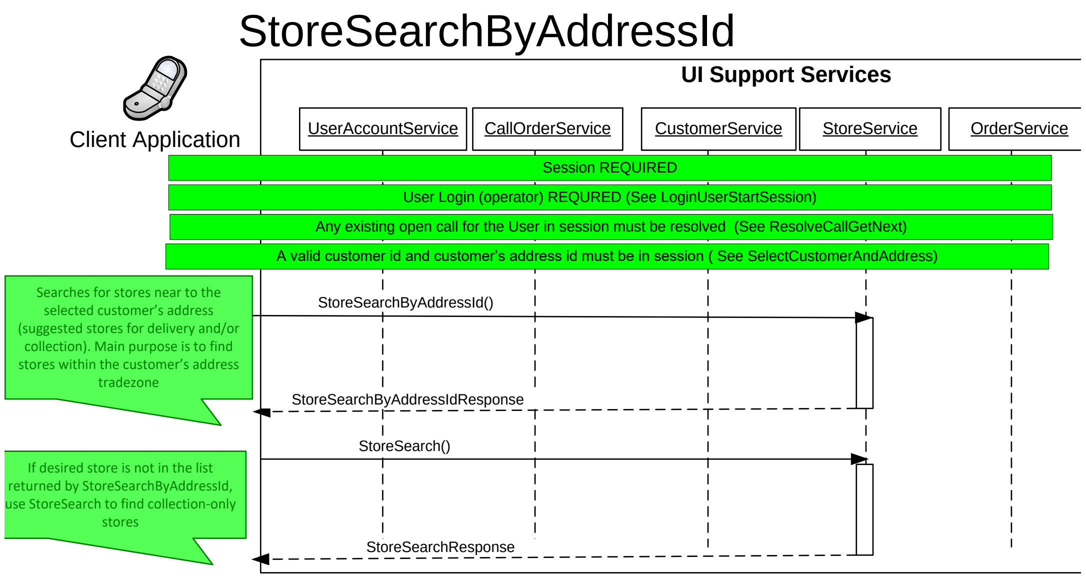
Customer Search


CustomerSearch
UI Support Services
Client Application
UserAccountService
CallOrderService
CustomerService
StoreService
OrderService
Session REQUIRED
User Login (operator) REQURED (See LoginUserStartSession)
Any existing open call for the User in session must be resolved (See ResolveCallGetNext)
CustomerSearch()
Searches for an existing customer upon criteria. Returns matching Customer's info and addresses
CustomerSearchResponse

Select a Customer and Address


SelectCustomerAndAddress
UI Support Services
Client Application
UserAccountService
CallOrderService
CustomerService
StoreService
OrderService
Session REQUIRED
User Login (operator) REQURED (See LoginUserStartSession)
Any existing open call for the User in session must be resolved (See ResolveCallGetNext)
SelectCustomerAndAddress()
Sets the customer id and address id to the current session. It also returns the selected customer's details upon flag in request.
SelectCustomerAndAddressResponse

Store Search


StoreSearchByAddressId
UI Support Services
Client Application
UserAccountService
CallOrderService
CustomerService
StoreService
OrderService
Session REQUIRED
User Login (operator) REQURED (See LoginUserStartSession)
Any existing open call for the User in session must be resolved (See ResolveCallGetNext)
A valid customer id and customer's address id must be in session (See SelectCustomerAndAddress)
Searches for stores near to the selected customer's address (suggested stores for delivery and/or collection). Main purpose is to find stores within the customer's address tradezone
StoreSearchByAddressId()
StoreSearchByAddressIdResponse
StoreSearch()
If desired store is not in the list returned by StoreSearchByAddressId, use StoreSearch to find collection-only stores
StoreSearchResponse

3 Service Contract Definitions

Service Method Definitions

The OHEICS security service is responsible for initiating a client session with the system.

Security Service

Specifically, the security service is used to begin and end a customer session across all OHEICS UI Support services.

StartSession

The StartSession method is used to initiate a customer session in the OHEICS system. This is similar to a web session and will expire after a pre-defined period of inactivity.

Note: This method must be called before calling any other web service method.

Input Parameters

Table 1 - StartSessionRequest

Type	Description
StartSessionRequest	A request object containing the information needed to initiate a customer session as well as the information needed for OHEICS to identify the calling application.

Output Parameters

Table 2 - StartSessionResponse

Type	Description
StartSessionResponse	A response object containing a customer session token to be passed to all subsequent service methods.

When to Use

This method should be called first before any other service methods.

This method should also be called whenever a service response is returned with a Service Error Type of SessionExpired and a new session is required.

Example: Start a New Session

Request

Call the StartSession method with the following:

- *StartSessionRequest*
    - *ApplicationId : "OHEICS-123"*
    - *CultureCode: "en-GB"*

Response

Receive the following response:

- *StartSessionResponse*
    - *StartSessionResultType: Success*
    - *Session*
        - *ApplicationId: "OHEICS-123"*
        - *CultureCode: "en-GB"*
        - *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*
    - *ErrorMessage: ""*
    - *ErrorType: None*
    - *IsSuccess: true*

EndSession

The EndSession method is used to finish a customer session in the OHEICS system and clear up any resources associated with this session. If this method is not called, the session will automatically expire after a pre-defined period of inactivity.

Input Parameters

Table 3 - SessionDTO

Type	Name	Description
SessionDTO	session	The session object to be expired on the server.

Output Parameters

This method does not return a value.

When to Use

This method should be called every time a customer finishes using the client application.

Example: End a Current Customer Session

Request

- *Session*
    - *ApplicationId: "OHEICS-123"*
    - *CultureCode: "en-GB"*
    - *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*

Store Service

The OHEICS store service is responsible for store information.

StoreSearch

Specifically, the StoreSearch method is used to locate the stores that are most appropriately placed to serve the customer based on their location and order requirements. This method will return a list of one or more stores, including details of the store opening times, store address and supported order types available at the store.

Input Parameters

Table 4 – StoreSearchRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
StoreSearchRequest	The store search request object contains customer location information and order information, such as order type and order date, used to find the nearest store(s).

Output Parameters

Table 5 - StoreSearchResponse

Type	Description
StoreSearchResponse	The return object contains a list of one or more stores.

When to Use

This method should be called as the first step in the ordering process after calling the StartSession method but before any calls to the OrderService.

Example: Find a Store

Request

- *Session*
    - *ApplicationId: “OHEICS-123”*
    - *CultureCode: “en-GB”*
    - *SessionId: “b1b9f875-79db-47e1-ad79-892e48a0a5c6”*
- *StoreSearchRequest*
    - *Address*
        - *PostcodeOrZip: “W39JF”*
    - *OrderTime: “29/10/2009 14:00”*

Response

- *StoreSearchResponse*
    - *ResultType: Success*
    - *StoreList[]*
        - *StoreDTO_0*
            - *DistanceInMilesFromSearchOrigin: 1.99958032929157*
            - *Email: "store@OHEICS.com"*
            - *OpeningHours*
                - *OpeningHours_0*
                    - *DayOfWeek: "Tuesday"*
                    - *OpeningPeriods*
                        - *OpengingTimePeriodDTO_0*
                            - *CloseTime: "23:00:00"*
                            - *OpenTime: "07:00:00"*
                - *OpeningHours_1*
                    - …
            - *StoreAddress*
                - *BuildingName: "2 St Andrews Court"*
                - *Latitude: 51.505751000000000*
                - *Longitude: -0.226078000000000*
                - *PostcodeOrZip: "W3 AL"*
                - *StreetName: "Wellington Street"*
                - *Territory: "Oxon"*
                - *TownCity: "Thame"*
            - *StoreId: 1*
            - *StoreName: "OHEICS Canteen"*
            - *TelephoneNumber: "000 00 000 000"*
    - *ErrorMessage: ""*
    - *ErrorType: None*
    - *IsSuccess: true*

When searching for delivery stores the AddressDTO will need to be specific enough to identify a single residence. Unlike searching for collection stores, the address will be used by the OHEICS engine to look for stores with a delivery area that will service the customer's home. Typically, a delivery address will only match a single store.

If the address is not specific enough, the service will return with a StoreSearchResultType of InvalidAddress.

Find a Store Using Longitude/Latitude

Stores can also be located using a customer's geo coordinates by passing latitude and longitude values as part of the AddressDTO within the StoreSearchRequest. In this case all other address values can be blank.

When using a customer's geo coordinates to search for a store, OHEICS will only be able to offer collection orders.

StoreSearchByIP

The StoreSearchByIP method is used to locate the stores that are most appropriately placed to serve the customer based on their IP address and order requirements. This method will return a list of one or more stores, including details of the store opening times, store address, and supported order types available at the store.

Input Parameters

Table 6 - StoreSearchByIP

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
StoreSearchByIP	The store search request object contains customer IP address and order information, such as order type and order date, used to find the nearest store(s).

Output Parameters

Table 7 - StoreSearchByIP

Type	Description
StoreSearchByIP	The return object contains a list of one or more stores.

When to Use

This method should be called as the first step in the ordering process after calling the StartSession method but before any calls to the OrderService.

Example: Find a Store

Request

- *Session*
  - *ApplicationId: "821D3E1C-8FA7-4974-814E-AD9BF27E4FA2"*
  - *CultureCode: "en-AU"*
  - *SessionId: "1e3a31d5-0970-4145-afbe-56b3eb8142dd"*
- *StoreSearchByIP*
  - *Address*

        - *IPAddress: "202.14.186.34"*
    - *OrderTime: "4/28/2014 2:27:34 PM"*

Response

- *StoreSearchByIP*
    - *ResultType: Success*
    - *StoreList[]*
        - *StoreDTO_0*
            - *DistanceInMilesFromSearchOrigin: 5.55594718458317*
            - *Email: "tampizza@bigpond.net.au"*
            - *OpeningHours*
                - *OpeningHours_0*
                    - *DayOfWeek: "Monday"*
                    - *OpeningPeriods*
                        - *OpengingTimePeriodDTO_0*
                            - *CloseTime: "00:00:00"*
                            - *OpenTime: "00:01:00"*
                - *OpeningHours_1*
                    - …
            - *StoreAddress*
                - *BuildingName: "Shop 2, 266-274"*
                - *Latitude: -31.0885*
                - *Longitude: 150.928*
                - *PostcodeOrZip: "2340"*
                - *StreetName: "Peel St"*
                - *Territory: "NSW"*
                - *TownCity: "Tamworth"*
            - *StoreId: 67*
            - *StoreName: "Tamworth"*
            - *TelephoneNumber: "1300 PIZZA HUT"*
    - *ErrorMessage: ""*
    - *ErrorType: None*
    - *IsSuccess: true*

GetStoreOrderOptions

This method gets the available store order options for starting an order at particular store.

Input Parameters

Table 8 - GetStoreOrderOptionsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetStoreOrderOptionsRequest	The request object contains the StoreId and the order time. Note that for the purposes of retrieving available order slots the order time provided will be the local store time.

Output Parameters

Table 9 - GetStoreOrderOptionsResponse

Type	Description
GetStoreOrderOptionsResponse	The response object containing information about possible order placement options.

When to Use

This method should be called once the customer has selected a store from those returned as part of the StoreSearch() response.

This method is required as a separate step in the process due to the potential intensity of each check for store order options. For example, if five stores have been found, and all implement the point of sale order slots retrieval mechanism, including this method within the store search would require five separate service requests to be required to five separate point of sale systems. As we cannot determine the response time for these calls this may take a considerable time.

By providing this method as a separate step in the order process we can minimize unnecessary processing, and retrieve information only for the selected store.

Example: Get Store Order Options for Store 1 from Local Time 29/10/2009 14:00

Request

- *Session*
    - *ApplicationId: "OHEICS-123"*
    - *CultureCode: "en-GB"*

- *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*
- *GetStoreOrderOptionsRequest*
  - *StoreId: 1*
  - *OrderTime: "29/10/2009 14:00"*

Response

- *GetStoreOrderOptionsResponse*
  - *ResultType: Success*
  - *StoreOrderOptions*
    - *StoreOrderOptionDTO_0*
      - *StoreId: 1*
      - *OrderClass: 1 (collection)*
      - *AvailableOrderSlots*
        - *DateTime_0: 29/10/2009 14:30*
        - *DateTime_1: 29/10/2009 14:45*
        - *DateTime_2: 29/10/2009 15:00*
        - *…..*
        - *DateTime_32: 29/10/2009 22:30*
        - *DateTime_33: 29/10/2009 22:45*
    - *StoreOrderOptionDTO_1*
      - *StoreId: 1*
      - *OrderClass: 2 (delivery)*
      - *AvailableOrderSlots*
        - *DateTime_0: 29/10/2009 15:00*
        - *DateTime_1: 29/10/2009 15:20*
        - *DateTime_2: 29/10/2009 15:40*
        - *…..*
        - *DateTime_20: 29/10/2009 21:40*
        - *DateTime_21: 29/10/2009 22:00*
  - *CollectNowAvailable: true*
  - *CollectNowPromiseTime: 29/10/2009 9:11:00 PM*
  - *CollectNowPromiseTimeSpan: 00:30:00*
  - *DeliverNowAvailable: false*
  - *DeliverNowPromiseTime: 1/1/0001 12:00:00 AM*
  - *DeliverNowPromiseTime: 00:00:00*
  - *BusinessDayDate: 29/10/2009 12:00:00 AM*
  - *ErrorMessage: ""*
  - *ErrorType: None*
  - *IsSuccess: true*

GetStoreList

This method is used to retrieve a list of stores sorted alphabetically. Since the list can be too big to display in a single screen of the user interface, it provides the calling application with the option to page the results. The caller can determine the size of each page (in number of stores) and the page number they want to be displayed.

Input Parameters

Table 10 - StoreListRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
StoreListRequest	The store list request object contains the size of the page in number of store, the page number we want returned and order date.

Output Parameters

Table 11 - StoreListResponse

Type	Description
StoreListResponse	The return object contains a list of one or more stores.

When to Use

This method should be called as the first step in the ordering process after calling the StartSession as an alternative to the StoreSearch method, but before any calls to the OrderService.

Example: Retrieve a List of Stores

Request

- *Session*
    - *ApplicationId: "OHEICS-123"*
    - *CultureCode: "en-GB"*
    - *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*
- *StoreListRequest*
    - *OrderTime: "29/10/2009 14:00"*
    - *PageNumber = 1*
    - *PageSize = 15*

Response

- *StoreSearchResponse*
    - *ResultType: Success*

- *StoreList[]*
    - *StoreDTO_0*
        - *DistanceInMilesFromSearchOrigin: 1.99958032929157*
        - *Email: "store@OHEICS.com"*
        - *OpeningHours*
            - *OpeningHours_0*
                - *DayOfWeek: "Tuesday"*
                - *OpeningPeriods*
                    - *OpengingTimePeriodDTO_0*
                        - *CloseTime: "23:00:00"*
                        - *OpenTime: "07:00:00"*
            - *OpeningHours_1*
                - …
        - *StoreAddress*
            - *BuildingName: "2 St Andrews Court"*
            - *Latitude: 51.505751000000000*
            - *Longitude: -0.226078000000000*
            - *PostcodeOrZip: "W3 AL"*
            - *StreetName: "Wellington Street"*
            - *Territory: "Oxon"*
            - *TownCity: "Thame"*
        - *StoreId: 1*
        - *StoreName: "OHEICS Canteen"*
        - *TelephoneNumber: "000 00 000 000"*
- *ErrorMessage: ""*
- *ErrorType: None*
- *IsSuccess: true*

GetStoreNames

This method is used to retrieve a list of store names in operation. This can be optionally filtered by region by passing in a region ID.

Input Parameters

Table 12 - StoreNamesRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

StoreNamesRequest	The store names request object contains an optional RegionId parameter to filter the stores returned by region.

Output Parameters

Table 13 - StoreNamesResponse

Type	Description
StoreNamesResponse	A response object containing the names and IDs of stores available nationally or within a selected region.

When to Use

A call to Get Store names may be used when just a list of store names and IDs are required. This might be for example when displaying a list for selection purposes.

Example: Get Store Names

Request

- *Session*
    - *ApplicationId: "OHEICS-123"*
    - *CultureCode: "en-GB"*
    - *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*
- *RegionId: 1*

Response

- *StoreNamesResponse*
    - *Stores[]*
        - *StoreId: 1*
        - *StoreName: "OHEICS Canteen"*

GetStoreDetail

This method is used to retrieve a full set of data for an individual store. The type of information provided includes store name details, store address details, opening hours and future data ordering configuration.

Input Parameters

Table 14 - StoreDetailRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

StoreDetailRequest	The store detail request object contains the StoreId of the required store and the week start date to retrieve the opening hours

Output Parameters

Table 15 - StoreDetailResponse

Type	Description
StoreDetailResponse	The store detail response object contains the Store detail and additional store meta data including the maximum number of days ahead orders can be placed at the store.

When to Use

A call to Get Store Detail may be used when comprehensive details of the store are required.

Example: Get Store Detail

Request

- *Session*
    - *ApplicationId: "OHEICS-123"*
    - *CultureCode: "en-GB"*
    - *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*
- *StoreDetailRequest*
    - *StoreId: 1*
    - *WeekStartDateTime: 2010-05-18T09:00:00*
    - *IncludeOrderTypeOpeningTimes: true*

Response

- *StoreDetailResponse*
    - *ResultType: Success*
    - *StoreDTO_0*
        - *Email: "store@OHEICS.com"*
        - *OpeningHours*
            - *OpeningHours_0*
                - *DayOfWeek: "Tuesday"*
                - *OpeningPeriods*
                    - *OpengingTimePeriodDTO_0*
                        - *CloseTime: "23:00:00"*
                        - *OpenTime: "07:00:00"*
                - *CollectionTimePeriods*

                - *OpeningTimePeriodDTO_0*
                    - *CloseTime:"23:00:00"*
                    - *OpenTime:"07:00:00"*
            - *DeliveryTimePeriods*
                - *OpeningTimePeriodDTO_0*
                    - *CloseTime:"23:00:00"*
                    - *OpenTime:"07:00:00"*
                    -
        - *OpeningHours_1*
                - *…*
    - *StoreAddress*
        - *BuildingName: "2 St Andrews Court"*
        - *Latitude: 51.505751000000000*
        - *Longitude: -0.226078000000000*
        - *PostcodeOrZip: "W3 AL"*
        - *StreetName: "Wellington Street"*
        - *Territory: "Oxon"*
        - *TownCity: "Thame"*
    - *StoreId: 1*
    - *StoreName: "OHEICS Canteen"*
    - *TelephoneNumber: "000 00 000 000"*
- *caErrorMessage: ""*
- *ErrorType: None*
- *IsSuccess: true*

GetStoreAttributeList

This method is used to retrieve a full set of store attributes for given brand

Input Parameters

Table 16 - StoreAtributeRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

StoreAttributeRequest_STORE_ATTRIBUTE_request	The store attribute list request object contains the brand identifier of the required store attribute and type of store attribute

Output Parameters

Table 17 - StoreAttributeResponse

Type	Description
StoreAttributeResponse	The store attribute response object contains the list of store attribute Data transfer objects.

When to Use

A call to Get Store attribute list may be used when full list of store attributes for a brand is required.

Example: Get Store Detail

Request

- *Session*
    - *ApplicationId: "OHEICS-123"*
    - *CultureCode: "en-GB"*
    - *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*
- *StoreAttributeRequest*
    - *BrandId: 1*
    - *StoreAttributeTypeId: 1*

Response

- *StoreAttributeResponse*
    - *StoreAttributeList[]*
        - *StoreAttributeDTO_0*
            - *StoreAttributeId: 1*
            - *StoreAttributeType: Services*
            - *DisplayText: "Delivery"*
            - *Description: "Delivery"*

StoreSearchByAddressId

This method will search for nearby stores using address ID as the search parameter; it can take the value either from session or provided in request.

Input Parameters

Table 18 - StoreSearchByAddressIdRequest

Type	Description
SessionDTO	The current session object
StoreSearchByAddressIdRequest	The request object containing the search parameters

Output Parameters

Table 19 - StoreSearchResponse

Type	Description
StoreSearchResponse	The return object contains a list of one or more stores.

When to Use

In a call center context, this method should be used to get the list of suggested stores for the customer (before calling this method, the customer and address should have been already selected, otherwise call SelectCustomerAndAddress method of CustomerService to do that).

This method will return collection stores and valid delivery stores (within the customer's address tradezoneId).

Example: Get Store Detail

Request

- *Session*
    - *ApplicationId: "OHEICS-123"*
    - *CultureCode: "en-GB"*
    - *SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"*
- *StoreSearchByAddressIdRequest*
    - *AddressId: null*
    - *AccountCode:null*
    - *IncludeOrderTypeOpeningTimes: true*

Response

- *StoreSearchResponse*
    - *ResultType: Success*
    - *StoreList[]*
        - *StoreDTO_0*
            - *DistanceInMilesFromSearchOrigin: 1.99958032929157*
            - *Email: "store@OHEICS.com"*
            - *OpeningHours*
                - *OpeningHours_0*
                    - *DayOfWeek: "Tuesday"*

-
  -
    -
      - ▪ *OpeningPeriods*
        - *OpengingTimePeriodDTO_0*
          - *CloseTime: "23:00:00"*
          - *OpenTime: "07:00:00"*
    - *OpeningHours_1*
      - ▪ …
  - StoreAddress
    - BuildingName: "2 St Andrews Court"
    - Latitude: 51.505751000000000
    - Longitude: -0.226078000000000
    - PostcodeOrZip: "W3 AL"
    - StreetName: "Wellington Street"
    - Territory: "Oxon"
    - TownCity: "Thame"
  - StoreId: 1
  - StoreName: "OHEICS Canteen"
  - TelephoneNumber: "000 00 000 000"
- ErrorMessage: ""
- ErrorType: None
- IsSuccess: true

AddStoreAsFavorite

This method will mark a store as favorite store.

Input Parameters

Table 20 - SaveStoreAsFavoriteRequest

Type	Description
SessionDTO	The current session object
SaveStoreAsFavoriteRequest	The request object containing the id of store

Output Parameters

Table 21 - SaveStoreAsFavoriteResponse

Type	Description
SaveStoreAsFavoriteResponse	The return object the success of saving store as a favorite store

When to Use

In current OHEICS, we only allow user to select a primary store for an address saved in the user's profile. In this SCR, we are going to allow the user to specify a store as a favorite.

Example: Get Store Detail

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "b1b9f875-79db-47e1-ad79-892e48a0a5c6"
- SaveStoreAsFavoriteRequest
    - StoreId: not null

Response

- SaveStoreAsFavoriteResponse
    - IsSuccess: true
    - ErrorMessage: ""
    - ErrorType: None

RemoveFavoriteStore

This method will remove a favorite store.

Input Parameters

Table 22 - SaveStoreAsFavoriteRequest

Type	Description
SessionDTO	The current session object
RemoveFavoriteStoreRequest	The request object containing the id of store

Output Parameters

Table 23 - RemoveFavoriteStoreResponse

Type	Description
RemoveFavoriteStoreResponse	The return object the success of removing store

When to Use

We are going to allow the user to remove a favorite store.

Example: Get Store Detail

Request

- Session

- ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “b1b9f875-79db-47e1-ad79-892e48a0a5c6”
- SaveStoreAsFavoriteRequest
    - StoreId: not null

Response

- RemoveFavoriteStoreResponse
    - IsSuccess: true
    - ErrorMessage: “”
    - ErrorType: None

GetCustomerFavoriteStoreList

This method will return a favorite stores list.

Input Parameters

Table 24 - GetCustomerFavoriteStoreListRequest

Type	Description
SessionDTO	The current session object
GetCustomerFavoriteStoreListRequest	The request object containing parameters.

Output Parameters

Table 25 - GetCustomerFavoriteStoreListResponse

Type	Description
SessionDTO	The current session object
GetCustomerFavoriteStoreListResponse	The return object contains a list of one or more favorite stores.

When to Use

We are going to return a list of favorite stores.

Example: Get Store Detail

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “b1b9f875-79db-47e1-ad79-892e48a0a5c6”
- GetCustomerFavoriteStoreListRequest

- NormalView: true
- OrderTime:DateTime.Now

Response

- GetCustomerFavoriteStoreListResponse
  - IsSuccess: true
  - StoreList[]
    - StoreDTO_0
      - DistanceInMilesFromSearchOrigin: 1.99958032929157
      - Email: “store@OHEICS.com”
      - OpeningHours
        - OpeningHours_0
          - DayOfWeek: “Tuesday”
          - OpeningPeriods
            - OpengingTimePeriodDTO_0
              - CloseTime: “23:00:00”
              - OpenTime: “07:00:00”
        - OpeningHours_1
          - …
      - StoreAddress
        - BuildingName: “2 St Andrews Court”
        - Latitude: 51.505751000000000
        - Longitude: -0.226078000000000
        - PostcodeOrZip: “W3 AL”
        - StreetName: “Wellington Street”
        - Territory: “Oxon”
        - TownCity: “Thame”
      - StoreId: 1
      - StoreName: “OHEICS Canteen”
      - TelephoneNumber: “000 00 000 000”
  - ErrorMessage: “”
  - ErrorType: None

Order Service

The OHEICS order service is responsible for all ordering functionality from getting a stores menu, to basket operations, and order placement.

GetMenuAndStartOrder

This method is used to load the entire menu at a selected store for a selected order type and order time.

Input Parameters

Table 26 - GetMenuAndStartOrderRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetMenuAndStartOrderRequest	A request object containing the information required to be able to start an order, including the store identifier, the order type and order time.

Output Parameters

Table 27 - GetMenuAndStartOrderResponse

Type	Description
GetMenuAndStartOrderResponse	A response object containing the entire menu hierarchy.

When to Use

This method should be called after the customer has selected a store using the StoreService. When a store is selected or if only one store is available (i.e. delivery orders) the customer should choose an order time from the list of available order times contained in the StoreDTO object. The client application will then have enough information to call GetMenuAndStartOrder method to load the menu at the store and initiate a shopping basket in the customer session on the server.

This method should also be called whenever the following order details change:

- Order time
- Order class (collection or delivery)

If the application allows the customer to change these details at any point in the order, this method will need to be called to update the server with the correct details.

Note: OHEICS supports different menus by order type and order time. For example, a store may have a breakfast-only menu or a different delivery menu. If this method is called to change order details, the menu returned may be different. In this case it may also be necessary to clear a customer's basket.

Example: Select to Order from the OHEICS Canteen Store for Collection at 22:15

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"

- SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - FulfillmentTimeType: 2
    - OrderClass: 1
    - OrderTime: “28/10/2009 22:15”
    - StoreId: 1

Response

- MenuRootCategory
    - CategoryDTO_0
        - Categories
            - Categories: null
            - CategoryClassId: 4
            - CategoryClassName: “Class 1”
            - CategoryId: 5
            - Description: “Sides”
            - ImageAltText: “Image alt text descriptions”
            - ImageUrl: “~/ImageRelativePath/ImageFileName.jpg”
            - MarketingDesription: “description”
            - Products
                - Product_0
                    - Description: “OHEICS's newest taste sensation”
                    - ImageAltText: “”
                    - ImageUrl: “~/ImageRelativePath/ImageFileName.jpg”
                    - MarketingDescription: “Description”
                    - Price: 6.00
                    - ProductId: 14
                    - SortOrder: 0
                    - SearchTerm: “”
                    - ValidityType: 1
        - CategoryClassId: -1
        - CategoryClassName: “”
        - CategoryId: -1
        - Description: “”
        - ImageAltText: “”
        - ImageUrl: “”
        - ItemUnavailableImageUrl: “”
        - MarketingDesription: “”

        - Products: null
- MenuId: 1
- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”

GetBasket

This method is used to load the current customer’s basket.

Input Parameters

Table 28 - GetBasketRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketRequest	A request object containing the information required to be able to retrieve the current basket.

Output Parameters

Table 29 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

It is likely that the calling application will cache and reuse its own basket object during the order process as service methods to modify the basket also return the current basket as a whole. Therefore this service method is more of a fallback call where the calling application cannot rely on its own data for basket information and requires a re-fetch of the current basket state.

The GetBasket() method does not modify the basket in any way.

Example: Get Details of the Current Basket

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”

        - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request

Note that currently the Request object is empty. All information about the current basket is held within the session management of OHEICS and is retrieved based on the provided SessionId property of the Session parameter

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: “Product1”
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00
            - ProductId: 1
    - NetItemTotal: 50.00
    - DeliveryChargeAmount: 5.00
    - DiscountAmount: 10:00
    - TaxChargeAmount: 7.50
    - BasketTotal: 52.50
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetBasketAddProduct

This method is used to add a specified product, wishlist, or previous order to the current basket and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 30 - GetBasketAddProductRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

GetBasketAddProductRequest	A request object containing the information required to be able to add a product, wishlist or previous order to the current basket.

Output Parameters

Table 31 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketAddProduct() method is called when the user selects a new menu item and adds it to their current basket. The method adds the product with the required quantity, calculates any discounts and deals, and returns the updated basket details to the calling application.

The method can also be used to add an existing customer wishlist or a previous order to the current basket. Note that the ProductType property of the request parameter can be used to define which type of Id the calling application is passing to the method, and therefore which type of object it should add to the basket.

Note also that the quantity property will only be used for menu product items. It is not applicable to the adding of wishlists or previous orders to the current basket.

Example: Add Menu Product with Id 2 to the Current Basket

Request

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
  - ProductTypeId: 2
  - Quantity: 2
  - ProductType: BasketProductAddType.ProductId
  - OrderTime: “2012-07-07 03:30 PM”
  - IsProductUpsellQualifier: false
  - IsProductUpSellOfferItem: true
  - ProductUpSellOfferId: 162
  - ProductUpSellOfferGroupId: 192372
  - ContainerStateDTO : Holds a list of modifiers that needs to be added to the basket.

        - ContainerStateANPDTO : Holds a list of modifiers in which add and no product to be added.

If a product contains a list of product modifiers and that need to be added to the basket all at once in Container model, then this DTO helps us to hold the modifiers information.

- ContainerStateDTO
    - ProductId : 4
    - Quantity : 1
    - ProductKey:2,4 (Product Key helps us to differentiate quantities for the product. For example if tomato is used in two different places in the hierarchy ProductKey helps us to get the correct quantity that been assigned.
    - ParentId: 2
    - ProductModifiers : Sub List of Product of Modifiers if the current product has related products.

There is a requirement to add ADD/NO product based on Default product selected or not selected.

We need to add ADD product in front of all product modifiers which is not default. And also if default product is not selected then we need to add no as well as the default product to basket. In order to accommodate the requirement, a new property is added to GetBasketAddProductRequest called ContainerStateANPDTO.

- ContainerStateANPDTO
    - ProductId : 4
    - Quantity : 1
    - ProductKey:2,4 (Product Key helps us to differentiate quantities for the product. For example if tomato is used in two different places in the hierarchy product key helps us to get the correct quantity that been assigned.
    - ParentId: 2
    - ProductModifiers : Sub List of Product of Modifiers if the current product has related products.
- If there is no requirement to add ADD/NO product this property can be set as null.

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: “Product1”
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00

- ProductId: 1
  - BasketItemDTO_1
    - BasketItemId: 2
    - Description: “Product2”
    - Price: 5.00
    - Quantity: 2
    - LineTotal: 10.00
    - ProductId: 2
  - NetItemTotal: 60.00
  - DeliveryChargeAmount: 5.00
  - DiscountAmount: 12:50
  - TaxChargeAmount: 9.00
  - BasketTotal: 61.50
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetBasketAddMultipleProducts

This method is used to add a collection of specified products, wishlist or previous order to the current basket and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 32 - GetBasketAddMultipleProductsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketAddMultipleProductsRequest	A request object containing the information required to be able to add a product, wishlist, or previous order to the current basket.

Output Parameters

Table 33 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketAddMultipleProducts() method is called when the user selects a collection of new menu items and adds them to their current basket. The method adds the products with the required quantity, calculates any discounts and deals etc and returns the updated basket details to the calling application.

The method can also be used to add an existing customer wishlist or a previous order to the current basket. Note that the ProductType property of the request parameter can be used to define which type of Id the calling application is passing to the method, and therefore which type of object it should add to the basket.

Note also that the quantity property will only be used for menu product items. It is not applicable to the adding of wishlists or previous orders to the current basket.

Example: Add Menu Products with Id 33,34 to the Current Basket

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-US”
    - SessionId: “e05ff78f-eaec-41bc-9ad2-84406f8e1caa”
- Request
    - Collection of products:
- {
- ProductTypeId: 243
- Quantity: 1
- PriceOverride: 1.50
- ProductType: BasketProductAddType.ProductId
- }
- {
- ProductTypeId: 244
- Quantity: 1
- PriceOverride: null
- ProductType: BasketProductAddType.ProductId
- }
-

        - OrderTime: "2013-06-17 05:30 PM"
        - ContainerStateDTO : Holds a list of modifiers that needs to be added to the basket.
        - ContainerStateANPDTO : Holds a list of modifiers in which add and no product to be added.

If a product contains a list of product modifiers and that needs to be added to the basket all at once in Container model then this DTO helps us to hold the modifiers information.

- ContainerStateDTO
    - ProductId : 4
    - Quantity : 1
    - ProductKey:2,4 (Product Key helps us to differentiate quantities for the product. For example if tomato is used in two different places in the hierarchy ProductKey helps us to get the correct quantity that been assigned.
    - ParentId: 2
    - ProductModifiers : Sub List of Product of Modifiers if the current product has related products.

There is a requirement to add ADD/NO product based on Default product selected or not selected.

We need to add ADD product in front of all product modifiers which is not default. And also if default product is not selected then we need to add no as well as the default product to basket. In order to accommodate the requirement a new property is added to GetBasketAddProductRequest called ContainerStateANPDTO.

- ContainerStateANPDTO
    - ProductId : 4
    - Quantity : 1
    - ProductKey:2,4 (Product Key helps us to differentiate quantities for the product. For example if tomato is used in two different places in the hierarchy product key helps us to get the correct quantity that been assigned.
    - ParentId: 2
    - ProductModifiers : Sub List of Product of Modifiers if the current product has related products.
- If there is no requirement to add ADD/NO product this property can be set as null.

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 13764
            - Description: "Pure Leaf Sweet Tea"

            - Price: 1.5
            - Quantity: 1
            - LineTotal: 1.5
            - ProductId: 243
        - BasketItemDTO_1
            - BasketItemId: 13765
            - Description: “Pure Leaf Tea Raspberry"
            - Price: 1.79
            - Quantity: 1
            - LineTotal: 1.79
            - ProductId: 244
    - NetItemTotal: 0.00
    - DeliveryChargeAmount: 0.00
    - DiscountAmount: 0.00
    - TaxChargeAmount: 0.00
    - BasketTotal: 3.39
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetBasketRemoveItem

This method is used to remove a specified basket item from the current basket and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 34 - GetBasketRemoveItemRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketRemoveItemRequest	A request object containing the information required to be able to remove an item from the current basket.

Output Parameters

Table 35 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketRemoveItem() method is called when the user selects an item in the current basket and removes it. The method removes the item and calculates any discounts and deals and returns the updated basket details to the calling application.

Example: Remove Basket Item with Id 2 from the Current Basket

Request

- Session
  - ApplicationId: "OHEICS-123"
  - CultureCode: "en-GB"
  - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
  - BasketItemId: 2

Response

- Basket
  - BasketId: 1
  - BasketItems
    - BasketItemDTO_0
      - BasketItemId: 1
      - Description: "Product1"
      - Price: 10.00
      - Quantity: 5
      - LineTotal: 50.00
      - ProductId: 1
  - NetItemTotal: 50.00
  - DeliveryChargeAmount: 5.00
  - DiscountAmount: 10:00
  - TaxChargeAmount: 7.50
  - BasketTotal: 52.50
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None

- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetBasketClearPartialBasket

This method is used to clear partial basket and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 36 - GetBasketRemoveItemRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketClearPartialBasket	A request object containing the information required to be able to clear partial basket

Output Parameters

Table 37 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketClearPartialBasket() method is called when the user clear partial basket. The method clear basket partially and returns the updated basket details to the calling application.

Example: Clear Partial Basket with Id 2 from the Current Basket

Request

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
  - inviteeId: 2

Response

- Basket

        - BasketId: 1
        - BasketItems
            - BasketItemDTO_0
                - BasketItemId: 1
                - Description: “Product1”
                - Price: 10.00
                - Quantity: 5
                - LineTotal: 50.00
                - ProductId: 1
        - NetItemTotal: 50.00
        - DeliveryChargeAmount: 5.00
        - DiscountAmount: 10:00
        - TaxChargeAmount: 7.50
        - BasketTotal: 52.50
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetBasketClearItems

This method is used to clear all items from current basket and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 38 - GetBasketClearItemsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketClearItemsRequest	A request object containing the information required to be able to clear all items from current basket

Output Parameters

Table 39 - GetBasketResponse

Type	Description

GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketClearPartialBasket() method is called when the user clear partial basket. The method clear basket partially and returns the updated basket details to the calling application.

Example: Clear Partial Basket with Id 2 from the Current Basket

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
    - inviteeId: 2

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: "Product1"
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00
            - ProductId: 1
    - NetItemTotal: 50.00
    - DeliveryChargeAmount: 5.00
    - DiscountAmount: 10:00
    - TaxChargeAmount: 7.50
    - BasketTotal: 52.50
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: ""
- Context: ""

- ValidationErrors: None

GetBasketUpdateQuantity

This method is used to update quantity of basket item in current basket and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 40 - GetBasketUpdateQuantityRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketUpdateQuantityRequest	A request object containing the information about basket item required to be able to update quantity from current basket

Output Parameters

Table 41 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketUpdateQuantity() method is called when the user changes quantity of basket item from current basket. The method update quantity of given basket item and returns the updated basket details to the calling application.

Example: Update Quantity with Id 1 &2 from the Current Basket

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - BasketItemsToUpdate_0
        - BasketItemId: 1

            - Quantity:2
    - BasketItemsToUpdate_1
        - BasketItemId: 2
        - Quantity:3

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: “Product1”
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00
            - ProductId: 1
    - NetItemTotal: 50.00
    - DeliveryChargeAmount: 5.00
    - DiscountAmount: 10:00
    - TaxChargeAmount: 7.50
    - BasketTotal: 52.50
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetBasketApplyVoucher

This method is used apply a specific voucher code and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 42 - GetBasketApplyVoucherRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

GetBasketApplyVoucherRequest	A request object containing the information required to be able to apply the voucher. This inherits from the GetBasketRequest and therefore contains all of its properties.

Output Parameters

Table 43 - GetBasketApplyVoucherResponse

Type	Description
GetBasketApplyVoucherResponse	A response object containing the current basket and a response status providing details of any problems with the request. This inherits from the GetBasketResponse and therefore contains all of its properties.

When to Use

During the ordering process, the GetBasketApplyVoucher () method is called when the user types in a voucher code and applies it. The method applies the voucher to the basket and that application stays with the basket until either the basket is destroyed (by session timeout, placing an order etc.) or the user decides to remove the voucher (this would call the GetBasketRemoveVoucher or RemoveVoucher method).

There are three possible outcomes to this method call:-

1. The voucher doesn't exist or is not valid for some reason. In which case the IsVoucherApplied property of the response would be false.
2. The voucher is valid but doesn't qualify (e.g. The voucher states that the user must spend £20 to qualify but they've only spent £10). In which case Response.IsVoucherApplied = true and Response.VoucherQualifies=false. Subsequent additions of product to the basket may change the Response.VoucherQualifies setting.
3. The voucher is valid and it qualifies. In which case Response.IsVoucherApplied = true and Response.VoucherQualifies=true.

Example: Add Voucher with a Code ABCD1234FT to the Basket. To Qualify Basket Total Must be 40.00

Request

- Session
  - ApplicationId: "OHEICS-123"
  - CultureCode: "en-GB"
  - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
  - VoucherCode: ABCD1234FT

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: “Product1”
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00
            - ProductId: 1
        - BasketItemDTO_1
            - BasketItemId: 2
            - Description: “Product2”
            - Price: 5.00
            - Quantity: 2
            - LineTotal: 10.00
            - ProductId: 2
    - NetItemTotal: 60.00
    - DeliveryChargeAmount: 5.00
    - DiscountAmount: 12:50
    - TaxChargeAmount: 9.00
    - BasketTotal: 61.50
- BasketResponseStatus: Success
- IsSuccess: true
- IsVoucherApplied: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- VoucherQualifies true
- VoucherCode – ABCD1234FT
- VoucherDescription – Spend £5 , get a half-price drink
- Voucherproducts – collection of half-price drinks which the user can choose from

GetBasketRemoveVoucher

This method is used remove a voucher code that has been previously applied and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 44 - GetBasketRemoveVoucherRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketRemoveVoucherRequest	A request object containing the information required to be able to apply the voucher. Currently, no information is required. This is reserved for future expansion. This inherits from the GetBasketRequest and therefore contains all of its properties.

Output Parameters

Table 45 - GetBasketRenoveVoucherResponse

Type	Description
GetBasketRenoveVoucherResponse	A response object containing the current basket and a response status providing details of any problems with the request. This inherits from the GetBasketResponse and therefore contains all of its properties.

When to Use

During the ordering process, the GetBasketRemoveVoucher () method is called when the removes a voucher code that was previously applied. The method unapplies the voucher from the basket. Any offer conditions that were added to the basket when the voucher was applied will be removed (e.g. discount or discounted products).

Example: Remove Voucher

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    -

Response

- Basket

        - BasketId: 1
        - BasketItems
            - BasketItemDTO_0
                - BasketItemId: 1
                - Description: “Product1”
                - Price: 10.00
                - Quantity: 5
                - LineTotal: 50.00
                - ProductId: 1
            - BasketItemDTO_1
                - BasketItemId: 2
                - Description: “Product2”
                - Price: 5.00
                - Quantity: 2
                - LineTotal: 10.00
                - ProductId: 2
        - NetItemTotal: 60.00
        - DeliveryChargeAmount: 5.00
        - DiscountAmount: 12:50
        - TaxChargeAmount: 9.00
        - BasketTotal: 61.50
    - BasketResponseStatus: Success
    - IsSuccess: true
    - IsVoucherApplied: false
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None
    - VoucherQualifies false

GetBasketAddVoucherProduct

This method is used to add any products to the basket that may have been offered at a discounted price as the result of a voucher being applied and the basket qualifying for the offer.

Input Parameters

Table 46 - GetBasketAddOfferRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketAddOfferRequest	A request object containing the information on the offer product that is to be added to the basket.

Output Parameters

Table 47 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketAddVoucherProduct () method is called after a voucher has been applied and one or more discounted products are on offer as a result of applying the voucher. After the voucher has been applied and the basket qualifies for the offer, the products on offer (if any) will be returned as part of the basket response in the VoucherProducts property. This is in the form of a collection of OfferProductDTO objects. These are always returned as a parent/child collection. The parent being the name of the group (e.g. Drinks) and the children of each group are also of type OfferProductDTO (e.g. Pepsi, Sprite etc.). The request needs to pass back the Group Id of the parent group and the Product ID of the OfferProduct on offer. Any other information in the VoucherProducts property is read-only and cannot be passed back the server.

Voucher products would normally be offered at a reduced price. Any reduced-price products that are added to the basket may not be modified and the server will ignore any such requests (e.g. if a half-price coke is added to the basket then it will not be possible to subsequently add a second one or to change the quantity of the item in the basket).

Example: Add Pepsi from Collection of Half-Price Drinks

Request

- Previous Response.Basket.VoucherProducts
    - Description: Drinks
    - ProductGroupId: 230
    - Children

            - Description: Coke
            - ProductGroupId: 231
            - ProductId: 1234
            - PreOffer price: 3:00
            - OfferPrice: 1.50
        - Description: Pepsi
            - ProductGroupId: 232
            - ProductId: 1235
            - PreOffer price: 3:00
            - OfferPrice: 1.50
- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
    - GroupId: 230
    - ProductId: 1235

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: "Product1"
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00
            - ProductId: 1
            - IsUpdateable: true
        - BasketItemDTO_1
            - BasketItemId: 2
            - Description: "Product2"
            - Price: 5.00
            - Quantity: 2
            - IsUpdateable:true
        - BasketItemDTO_3
            - BasketItemId: 3
            - Description: "Pepsi"
            - ProductId: 1235

                - Price: 1.50
                - Quantity: 1
                - IsUpdateable: false
            - LineTotal: 10.00
        - NetItemTotal: 61.50
        - DeliveryChargeAmount: 5.00
        - BasketTotal: 61.50
- BasketResponseStatus: Success
- IsSuccess: true
- IsVoucherApplied: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- VoucherQualifies:true

GetBasketAddUpsellProducts

This method is used to add any products to the basket that may have been offered at a discounted price as the result of an upsell offer that the basket qualifies for.

Input Parameters

Table 48 - GetBasketAddOfferRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketAddOfferRequest	A request object containing the information on the offer product that is to be added to the basket.

Output Parameters

Table 49 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the GetBasketAddUpSellProduct () method is called if the basket applies for a particular UpSell offer. If the basket qualifies for the offer, the products on offer (if any) will be returned as part of the basket response in the UpSellProducts property. This is in the form of a collection of OfferProductDTO objects. These are always returned as a parent/child collection. The parent being the name of the group (e.g. Drinks) and the children of each group are also of type OfferProductDTO (e.g. Pepsi, Sprite etc.). The request needs to pass back the Id of the parent group and the Product ID of the OfferProduct on offer. Any other information in the UpSellProducts property is read-only and cannot be passed back the server.

UpSell products may or may not be offered at a reduced price. Any reduced-price products that are added to the basket may not be modified and the server will ignore any such requests (e.g. if a half-price coke is added to the basket then it will not be possible to subsequently add a second one or to change the quantity of the item in the basket). However, any products that are added at full price may be modified.

Example: Add Pepsi from Collection of Half-Price Drinks

Request

- Previous Response.Basket.UpSellProducts
    - Description: Drinks
    - ProductGroupId: 230
    - Children
        - Description: Coke
        - ProductGroupId: 231
        - ProductId: 1234
        - PreOffer price: 3:00
        - OfferPrice: 1.50
    - Description: Pepsi
        - ProductGroupId: 232
        - ProductId: 1235
        - PreOffer price: 3:00
        - OfferPrice: 1.50
- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - GroupId: 230
    - ProductId: 1235

Response

- Basket

- BasketId: 1
- BasketItems
    - BasketItemDTO_0
        - BasketItemId: 1
        - Description: “Product1”
        - Price: 10.00
        - Quantity: 5
        - LineTotal: 50.00
        - ProductId: 1
        - IsUpdateable:true
    - BasketItemDTO_1
        - BasketItemId: 2
        - Description: “Product2”
        - Price: 5.00
        - Quantity: 2
        - LineTotal: 10.00
        - ProductId:2
        - IsUpdateable:true
    - BasketItemDTO_3
        - BasketItemId: 3
        - Description: “Pepsi”
        - ProductId: 1235
        - Price: 1.50
        - Quantity: 1
        - Line Total: 1.50
        - IsUpdateable: false
- NetItemTotal:61.50
- DeliveryChargeAmount: 5.00
- BasketTotal: 66.50

- BasketResponseStatus: Success
- IsSuccess: true
- IsVoucherApplied: false
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- VoucherQualifies: false

SaveBasketAsWishList

This method is used to create favorites based on the current basket.

Input parameters

Input Parameters

Table 50 - SaveBasketAsWishlistRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
SaveBasketAsWishlistRequest	The request object. This object implements the base request and does not contain any additional information.

Output Parameters

Table 51 - SaveBasketAsWishlistResponse

Type	Description
SaveBasketAsWishlistResponse	A response object containing the base response. (No additional information specific to this operation).

When to Use

This method is called when the user needs to store a favorite selection based on their current basket. The user must have an active basket (see operations GetMenuAndStartOrder() and GetBasket()).

Example

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request

Response

- IsSuccess: true
- ErrorType: None

GetOrderConfirmation

This method is used to order confirmation details.

Input Parameters

Table 52 - GetOrderConfirmationRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetOrderConfirmationRequest	This object is empty right now but can be expanded in future.

Output Parameters

Table 53 - GetOrderConfirmationResponse

Type	Description
GetOrderConfirmationResponse	A response object containing the Order, Store and Order confirmation details.

When to Use

This method is called just after confirming order to get order confirmation details i.e. Order confirmation number, store and other order details. This function can be called to get data for the display order confirmation page, to show order confirmation number, from which store user has ordered, and store contact details in case the user wants to contact the store for any query.

Example

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request

Response

- Order
    - OrderId: 1001
    - OrderTotal: 20.00
    - DiscountAmount: 2.0
    - DeliveryChargeAmount: 3.0

- TaxChargeAmount: 2.0
- OrderReference: “REF23443”
- FulfillmentTimeTypeText: “”
- DateOrderRequired: 20/06/2012 12:23:000
- OrderMethod: “Online”
- VoucherCode: “”DIS10PER”
- VoucherDescription: “10% discount”
- OrderItems
  - OrderItemDTO_0
    - OrderItemId: 123
    - ProductId: 2345
    - ProductText: “Pepsi1Ltr”
    - ProductDescription: “Pepsi 1 Ltr”
    - Quantity: 2
    - Price: 1.2
    - PriceTotal: 2.4
    - IsNoChargeItem: false
    - DisplayOrder: 1
    - PrimarySku: 1001
    - IsSurcharge: false
    - SurchargeType: NOT_SET
    - SurchargeName: ””
- OrderPayments
  - OrderPaymentDTO_0
    - OrderPaymentId: 342
    - PaymentMethod: Cash
    - AmountTotal: 25.0

- Store
  - DistanceInMilesFromSearchOrigin: 1.99958032929157
  - Email: “store@OHEICS.com”
  - OpeningHours
    - OpeningHours_0
      - DayOfWeek: “Tuesday”
      - OpeningPeriods
        - OpengingTimePeriodDTO_0
          - CloseTime: “23:00:00”
          - OpenTime: “07:00:00”
    - OpeningHours_1
      - …

    - StoreAddress
        - BuildingName: “2 St Andrews Court”
        - Latitude: 51.505751000000000
        - Longitude: -0.226078000000000
        - PostcodeOrZip: “W3 AL”
        - StreetName: “Wellington Street”
        - Territory: “Oxon”
        - TownCity: “Thame”
    - StoreId: 1
    - StoreName: “OHEICS Canteen”
    - TelephoneNumber: “000 00 000 000”
- CustomerEmailAddress: emailed@domain.com
- CustomerPhoneNumber: “0123456789”
- CustomerFirstName: “Jim”
- CustomerLastName: “Colin”
- OrderSuccess: true
- OrderStatusMessage: “Order status message”
- CustomerId: 23423

ValidateBasketAgainstChanges

This method is used to refresh the basket when there is a change in store or order time and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 54 – ValidateBasketAgainstChangesRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
ValidateBasketAgainstChangesRequest	A request object containing the information required to be able to refresh a basket.

Output Parameters

Table 55 - ValidateBasketAgainstResponse

Type	Description
ValidateBasketAgainstResponse	A response object containing the current basket and a response status providing details of any problems with the request.

When to Use

During the ordering process, the ValidateBasketAgainstChanges()method is called to refresh the user's current basket when there is a change to the time or a store there is a chance that the user's current basket will be emptied and the user will be forced to start over again causing frustration for the user. Depend on product restriction, exclusion and voucher validation, the method keeps or remove the products in the current basket with quantity, calculates any discounts and deals etc and returns the updated basket details to the calling application.

Example: Validate Basket When Changing Store: Store1 to Store2

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-US"
    - SessionId: "e05ff78f-eaec-41bc-9ad2-84406f8e1caa"
- Request
    - OrderTime: null
    - FulfillmentTimeType: null
    - StoreId: 1.

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 13764
            - Description: "Pure Leaf Sweet Tea"
            - Price: 1.5
            - Quantity: 1
            - LineTotal: 1.5
            - ProductId: 243
        - BasketItemDTO_1
            - BasketItemId: 13765

        - Description: “Pure Leaf Tea Raspberry"
        - Price: 1.79
        - Quantity: 1
        - LineTotal: 1.79
        - ProductId: 244
    - NetItemTotal: 0.00
    - DeliveryChargeAmount: 0.00
    - DiscountAmount: 0.00
    - TaxChargeAmount: 0.00
    - BasketTotal: 3.39
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- IsReApplyVoucher: false
- IsReApplyVoucherSucess: false

ConfirmValidateBasketAgainstChanges

This method is used to confirm the changes (change store, order time, order type) users made or to revert the changes to the previous state. For example, users choose store A to order, then they change the store to Store B. Then the FrontEnd need to call ConfirmValidateBasketAgainstChanges, there are two options:

1. Confirm: users confirm the changes, from new they will order from Store B
2. Revert: users don’t want to choose store B, back to store A to order

Input Parameters

Table 56 - ConfirmValidateBasketAgainstChangesRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
ConfirmValidateBasketAgainstChangesRequest	A request object containing the information required to confirm or revert changes

Output Parameters

Table 57 - ConfirmValidateBasketAgainstChangesResponse

Type	Description
ConfirmValidateBasketAgainstChangesResponse	A response object containing the current basket, new order time, storied

When to Use

When a FrontEnd call ValidateBasketAgainstChanges with the DataMember IsRevertPossible.

Of ValidateBasketAgainstChangesRequest is true, ConfirmValidateBasketAgainstChanges must be called right after the ValidateBasketAgainstChanges call.

Example: Confirm ValidateBasket When Changing Store: Store1 to Store2

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-US”
    - SessionId: “e05ff78f-eaec-41bc-9ad2-84406f8e1caa”
- Request
    - RollbackChanges : false

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 13764
            - Description: "Pure Leaf Sweet Tea"
            - Price: 1.5
            - Quantity: 1
            - LineTotal: 1.5
            - ProductId: 243
        - BasketItemDTO_1
            - BasketItemId: 13765
            - Description: “Pure Leaf Tea Raspberry"
            - Price: 1.79
            - Quantity: 1
            - LineTotal: 1.79

                - ProductId: 244
        - NetItemTotal: 0.00
        - DeliveryChargeAmount: 0.00
        - DiscountAmount: 0.00
        - TaxChargeAmount: 0.00
        - BasketTotal: 3.39
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- StoreId: Id of Store 2
- OrderTime: new valid order timeREVERT VALIDATEBASKET WHEN CHANGING STORE: STORE1 to STORE2

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-US”
    - SessionId: “e05ff78f-eaec-41bc-9ad2-84406f8e1caa”
- Request
    - RollbackChanges : true

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 13764
            - Description: "Pure Leaf Sweet Tea"
            - Price: 1.5
            - Quantity: 1
            - LineTotal: 1.5
            - ProductId: 243
        - BasketItemDTO_1
            - BasketItemId: 13765
            - Description: “Pure Leaf Tea Raspberry"
            - Price: 1.79
            - Quantity: 1
            - LineTotal: 1.79

        - ProductId: 244
    - NetItemTotal: 0.00
    - DeliveryChargeAmount: 0.00
    - DiscountAmount: 0.00
    - TaxChargeAmount: 0.00
    - BasketTotal: 3.39
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- StoreId: Id of Store 1
- OrderTime: valid order time

GetBucketConfigurator

This method is used to configure bucket like products. These products comprise of the root product containing the overall description, price and the validity of the current configuration. Below this root product are a set of steps which themselves contain child step items. The steps contain information about the step and the configuration parameters (e.g. whether step is mandatory). The step items contain product option information and information about the configuration (e.g.MaxRequiredQuantity).

Input Parameters

Table 58 - GetBucketConfiguratorRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBucketConfiguratorRequest	A request object containing the root product ID, the basket item ID and quantity.

Output Parameters

Table 59 - GetBucketConfiguratorResponse

Type	Description

GetBucketConfiguratorResponse	A response object containing the root bucket product which itself holds the configuration state.

When to Use

This method is called initially when selecting a bucket type product e.g. “10 Piece Bargain Bucket”. The root product id is used to return the default configuration for the selection which is returned in the ConfigurationState property of the root product.

If a basket item ID is passed to this operation, the configuration returned is that which is currently in the basket for the selected product.

Example: 12 Piece Family Feast

The requests and responses associated with this call can be found in the appendix.

After starting a new session, retrieving store and menu details a call is made to GetBucketConfigurator.

Request

The request passes in standard basic request information (e.g. ApplicationId, culture code and SessionId) and the root product identifier.

Response

The response returns a success indicator of the call and then the root product details. The root product details also contains the steps in bucket configuration which themselves contain a number of child items within the steps.

The root product also contains the ConfigurationState. This object contains the state of the current configuration which must be used in subsequent calls to configure the bucket as UI Services does not maintain the state of the configuration.

CompleteBucketConfiguration

This method is used to finalize the configuration and insert the selected items into the basket.

Input Parameters

Table 60 - CompleteBucketConfiguratorRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
CompleteBucketConfiguratorRequest	A request object containing the current configuration state and any applicable BasketItemId.

Output Parameters

Table 61 - GetBucketConfiguratorResponse

Type	Description
GetBucketConfiguratorResponse	A response object containing the root bucket product which itself holds the configuration state.

When to Use

This call is typically made when the user has made all the desired and necessary updates to the bucket from previous calls to UpdateBucketItem and has arrived at a valid configuration.

This call will finalize the selection and add the root product and its configuration to the basket.

If a BasketItemId has been passed in the request, then the call is deemed as an update to an existing selection in the basket and therefore an additional product of this same type will not be added to the basket, rather the existing item will be updated.

Example: Complete Configuration for 12-Piece Family Feast

Request

The request contains the current configuration state and an optional BasketItemId

Response

The response object is of type 'GetBucketConfiguratorResponse'. This is the same type of object which is returned from the call to GetBucketConfigurator.

GetConfigurator

This method is used to configure a product with bespoke selections, building on top of a default configuration. So for example when a pizza product is selected it comprises of a default configuration (e.g. thin and crispy 7" base, tomato sauce, mozzarella cheese, ham and pineapple). Product configuration enables the customer to modify the default configuration to add or remove desired ingredients. Each call to this method is made with a ProductionConfigurationAction which determines which part of the product to configure and also defines the start and end points in the product configuration process.

Input Parameters

Table 62 - GetConfiguratorRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

GetConfiguratorRequest	A request object containing product Ids and the selected ProductConfigurationAction.

Output Parameters

Table 63 - GetConfiguratorResponse

Type	Description
GetConfiguratorResponse	A response object containing the current product configuration state and the available options in the context of the current state.

When to Use

This method is called within the context of a selected store (see StoreSearch), after starting an order and retrieving a menu (see GetMenuAndStartOrder). The menu returned in the latter call will contain a list of products. The products may or may not contain a value for 'CustomConfigurationAlias'. Those which have a value for this property can be configured further with custom selections.

The first call to GetConfigurator is made with ProductConfigurationAction type of 'ComplexProductSelected'. This starts of the process of configuration. Subsequent calls to GetConfigurator will configure the product selected by modifying its ingredients.

Each call to GetConfigurator returns in its response the current state of the configuration and the options for making further configuration changes to the product. The state object comprises of product/ingredient identifiers and lists of selected ingredients. The configuration options object in the response comprises of a list of available ingredients to select as a change to the current configuration.

The final call to GetConfigurator will be to either complete or cancel the configuration with ProductConfigurationActions of type CompleteConfiguration and CancelConfiguration respectively.

CompleteConfiguration will add the configured product to the current basket. CancelConfiguration will remove the current configuration from session state

Example: BBQ Chicken – Large pizza

The full chain of requests and responses for this example can be found in the appendix.

In this example, the user selects a "BBQ Chicken" pizza product to configure. It is assumed that the user has selected a store, retrieved a menu and started the order by calling the following service operations:

- StartSession()
- GetMenuAndStartOrder()
- GetBasket()

The first call to GetConfigurator() is made with a ProductConfigurationAction of ComplexProductSelected and a product id of 38 (BBQ Chicken). The response will contain the configuration state and configuration options.

The configuration state will contain the default configurations for the product selected. The configuration options will contain the available ids which can be used in subsequent GetConfigurator calls to further configure the product.

Request #1 - complexproductselected

The initial call to GetConfigurator() (with a ProductConfigurationAction of ComplexProductSelected) is made to select the initial product to configure (product 38 – BBQ Chicken). The product Id and other information about the products would have been available in the previous call to GetMenuAndStartOrder(). Those products which can be configured by the customer further would have a value for 'CustomConfigurationAlias'.

Response #1

The response contains default ingredient selections for the selected product and configurable options in the ConfigurationState and ConfigurationOptions xml response nodes respectively. The default selections under ConfigurationState are mostly single ids for various selection items, e.g. SelectedBaseTypeId, SelectedSizeId etc (full information for these various selected items can be found in the ConfigurationOptions node of the response). The default selections for toppings are contained in the array of SelectedToppings.

Request #2

The 2nd call to GetConfigurator() is to select an additional topping. Toppings are keyed on a composition of three ids, the ToppingId, the ParentProductToppingGroupId and the ProductClassId. In order to select a topping the request properties Id, ParentProductToppingGroupId and ClassId must be populated to denote which topping is required. The topping selected in the example is for 'Semi Dried Tomato (single)'. A full explanation of the data structure of toppings can be found in the 'Toppings'**Error! No bookmark name given.** section below.

Response #2

The toppings in the SelectedToppings child of the ConfigurationState node of the response now contains the detail of the topping selected in the request. This topping is no longer included in the list of available toppings in the Toppings child of the ConfigurationOptions node of the response.

Request #3 – completeconfiguration

The final call to GetConfigurator() is to complete the configuration. This will add the configured product to the current basket. An updated view of the basket can be seen by making a call to GetBasket().

Response #3 – completeconfiguration

The response object has reset the ConfigurationState / ConfigurationOptions to nil values.

ConfigureProduct

This method is used to load complex product and configure based on given request configuration state for each section. So for example it loads pizza product it comprises of a default configuration (e.g. thin and crispy 7" base, tomato sauce, mozzarella cheese, ham and pineapple). Product configuration enables the customer to modify the default configuration to add or remove desired ingredients. Add or remove ingredients can be done via configuration selections for each section.

Input Parameters

Table 64 - ConfigureProductRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
ConfigureProductRequest	Contains product id to load, and configuration to modify default configuration. i.e. change specialty, size, base, toppings etc.

Output Parameters

Table 65 - ConfigureProductResponse

Type	Description
ConfigureProductResponse	A response object containing current configuration state, validation result for validation errors and list of successful actions performed.

When to Use

This method is called when we want to load product with pre-selected configuration changes, especially in deal steps when users go back to previous steps and to make changes in pre-selected pizzas. This will enable users to retain the last selected configuration and reload it in session state in services.

GetDealConfigurator

This method is used to configure a deal with bespoke selections, building on top of a default configuration. So for example when a deal is selected it comprises of deal steps (e.g. choose a pizza, a drink, and choose a side). Deal configuration enables the customer to go through deal steps and add or remove desired products from available options. Each call to this method is made with a ProductionConfigurationAction which determines which part of the product to configure and also defines the start and end points in the product configuration deal steps.

Input Parameters

Table 66 - GetDealConfigurationRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetDealConfigurationRequest	A request object containing product Id for deal quantity.

Output Parameters

Table 67 - GetDealConfigurationResponse

Type	Description
GetDealConfigurationResponse	A response object containing the current product configuration state for the deal and the available options in the context of the current state.

When to Use

This method is called within the context of a selected store (see StoreSearch), after starting an order and retrieving a menu (see GetMenuAndStartOrder). The menu returned in the latter call will contain a list of products. The products may or may not contain a value for 'CustomConfigurationAlias'. Those which have a value for this property can be configured further with custom selections.

Example

Request

- Session
  - ApplicationId: "OHEICS-123"
  - CultureCode: "en-GB"
  - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
  - ProductId: 123
  - Quantity: 1

Response

- DealSteps
  - DealConfigurationDTO_0
    - ConfigurationCurrentStep: 1
    - ConfigurationRootproductName: "Pizza_Side_Combo"
    - ConfigurationRootProductDisplayDescription: "Large pizza and a side dish"

        - ConfigurationRootProductMarketingText: “Buy choice of large of pizza and side dish for £5.30”
        - DealSteps
            - DealStepConfigurationDTO_0
                - ProductId: 3445
                - Description: “Large pizza”
                - rootCategoryid: 3223
                - BasketItemId: 23
                - AllowedProductIds
                    - 123
                    - 2344
            - …
- SelectedProducts
    - ProductConfigurationKeyValuePairDTO_0
        - ProductId: 2345
        - Configuration
            - Configuration will be loaded as per product selected

DealCompleted

This method completes the deal and saves to the current user’s basket.

Input Parameters

Table 68 - DealCompletedRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
DealCompletedRequest	A request object inherited from GetBasketRequest object and contains details about deal to be completed.

Output Parameters

Table 69 - DealCompletedResponse

Type	Description
DealCompletedResponse	Response object inherit all its member from GetBasketResponse, it can be expanded in future.

When to Use

This method is called within the context of a current deal. After product selection through deal steps when user wants to add selected deal to basket this method needs to be called.

Example: Add Current Deal to Basket and Get Details of the Current Basket

Request

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
  - IsCancelled: false
  - Quantity: 1
  - SelectedProducts
    - productConfigurationKeyValuePairDTO_0
      - ProductId:234
      - Configuration: productconfiguration
      - ContainerState: current product container state
      - BasketItemId:null
  - BasketitemId:null
  - IsProductUpsellQualifier: false
  - IsProductUpSellOfferItem: true
  - ProductUpSellOfferId: 162
  - ProductUpSellOfferGroupId: 192372

Response

- Basket
  - BasketId: 1
  - BasketItems
    - BasketItemDTO_0
      - BasketItemId: 1
      - Description: “Product1”
      - Price: 10.00
      - Quantity: 5
      - LineTotal: 50.00
      - ProductId: 1
  - NetItemTotal: 50.00
  - DeliveryChargeAmount: 5.00
  - DiscountAmount: 10:00

    - TaxChargeAmount: 7.50
    - BasketTotal: 52.50
- BasketResponseStatus: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

ApplyVoucher

This method is used to apply a specific voucher code and update the basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 70 - ApplyVoucherRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
ApplyVoucherRequest	A request object containing the voucher code to apply on current basket.

Output Parameters

Table 71 - ApplyVoucherResponse

Type	Description
ApplyVoucherResponse	A response object contains list of validation messages and terms and conditions details and other attributes for voucher.

When to Use

During the ordering process, the ApplyVoucher () method is called when the user types in a voucher code and applies it. The method applies the voucher to the basket and that application stays with the basket until either the basket is destroyed (by session timeout, placing an order etc.) or the user decides to remove the voucher (this would call the GetBasketRemoveVoucher or RemoveVoucher method).

There can be three possible outcomes to this method call:-

1. The voucher doesn’t exist or is not valid for some reason.
2. The voucher is valid but doesn’t qualify (e.g. the voucher states that the user must spend £20 to qualify but they’ve only spent £10).

3. The voucher is valid and it qualifies. In which case voucher is applied to basket and updated.

Example: Apply Voucher with Code ABCD1234FT to the Basket

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
    - VoucherCode: ABCD1234FT

Response

- VoucherCode:ABCD1234FT
- VoucherDescription: "Brief description of voucher"
- IsVoucherValid: true
- VoucherEnabled: true
- VoucherActive: true
- TermsAndConditionsText: "Detailed terms and conditions for this voucher"
- TermsAndConditionsUrl: "http://domain.com/Voucher/TermsAndConditions.html"
- DisplayMessages
    - null

RemoveVoucher

This method is used remove a voucher code that has been previously applied, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 72 - RemoveVoucherRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
RemoveVoucherRequest	A request object containing the information required to be able to remove the voucher from current basket. This request object is currently empty because there can be only one voucher can be applied to basket.

Output Parameters

Table 73 - RemoveVoucherResponse

Type	Description
RemoveVoucherResponse	Response object is empty now.

When to Use

During the ordering process, the RemoveVoucher () method is called when the removes a voucher code that was previously applied. The method unapplies the voucher from the basket. Any offer conditions that were added to the basket when the voucher was applied will be removed (e.g. discount or discounted products).

Example: Remove Voucher

Request

- Session
  - ApplicationId: "OHEICS-123"
  - CultureCode: "en-GB"
  - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
  - VoucherCode:"Voucher-Code"

Response

- Response
  - IsSuccess:True/false
  - ErrorType: [None/ GeneralError/ AuthenticationError/ SessionExpired/ DataAccessError]
  - ErrorMessage: "This is error message"
  - ValidationErrors:
    - ErrorCode: "err-code"
    - ErrorDescription: "error Description"
  - IsOrderClosedinPos: True/false

ResetSessionOrderDetails

This method is used to reset order details as they are reset after placing order. Function create new order state and add to session ready to accept next order details.

Table 74 - ResetSessionOrderDetailsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
ResetSessionOrderDetailsRequest	This request object is empty now.

Output Parameters

Table 75 - ResetSessionOrderDetailsResponse

Type	Description
ResetSessionOrderDetailsResponse	Response object is empty now.

When to Use

During the ordering process, this function can be called to reset order details in session state. This function will remove all items from basket, any vouchers applied will be removed. New empty basket will be created and added to session state.

Example: Remove Voucher

Request

- Session
  - ApplicationId: "OHEICS-123"
  - CultureCode: "en-GB"
  - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
  - Context:""

Response

- Response
  - IsSuccess:True/false
  - ErrorType: [None/ GeneralError/ AuthenticationError/ SessionExpired/ DataAccessError]
  - ErrorMessage: "This is error message"
  - ValidationErrors:
    - ErrorCode: "err-code"
    - ErrorDescription: "error Description"
  - IsOrderClosedinPos: True/false

GetOrderPromiseTime

This method is used to get promise time for current order from selected store.

Input Parameters

Table 76 - GetOrderPromiseTime

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

Output Parameters

Table 77 - Int

Type	Description
Int	Returns order promise time in minutes.

When to Use

This function is called to get the order promise time for the current order in session.

Example: Get Order Promise Time

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”

Response

- Int: 15

GetComplexOrderPromiseTime

This method is used to get promise time for current order from selected store.

Input Parameters

Table 78 - GetComplexOrderPromiseTime

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

Output Parameters

Table 79 - GetComplexOrderPromiseTimeResponse

Type	Description
GetComplexOrderPromiseTimeResponse	A response object containing the entire promise time details based on the category.

When to Use

This function is called to get order promise time for current order in session. It uses the productweight, orderamount, posload and the schedule into consideration while calculating the promisetime.

Example: Get Complex Order Promise Time

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"

Response

- DateOrderRequired: 2012-7-7 17:30
- IsComplexPromiseTimeEnabled: true
- PromiseTime: 30
- OffsetTime: 25
- PromiseTimeByWeight: 0
- PromiseTimeByAmount: 10
- PromiseTimeByPOSLoad: 20
- PromiseTimeBySchedule: 0

GetCategoryAndStartOrder

This method is used to load the entire menu at a selected store for a selected order type and order time.

Input Parameters

Table 80 - GetCategoryAndStartOrderRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

GetCategoryAndStartOrderRequest	A request object containing the information required to be able to start an order, including the store identifier, the order type and order time.

Output Parameters

Table 81 - GetCategoryAndStartOrderResponse

Type	Description
GetCategoryAndStartOrderResponse	A response object containing the entire menu hierarchy.

When to Use

This method should be called after the customer has selected a store using the StoreService. This is similar to GetMenuAndStartOrder. The main difference is that it returns only the category and subcategory details.

For on-demand loading the front-end should invoke the methods in the following sequence:

1. GetCategoryAndStartOrder: To get the list of categories and subcategories.
2. GetProductsByCategory: To get the products for the selected category.
3. GetProductByProductId: To get the product details for the selected product.

Example: Getcategoryandstartorder

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - FulfillmentTimeType: 2
    - OrderClass: 1
    - OrderTime: “28/10/2009 22:15”
    - StoreId: 1
    - RemoveEmptyCategory: true

Response

- MenuRootCategory
    - CategoryDTO_0
        - Categories
            - Categories: null
            - CategoryClassId: 4

        - CategoryClassName: “Class 1”
        - CategoryId: 5
        - Description: “Sides”
        - ImageAltText: “Image alt text descriptions”
        - ImageUrl: “~/ImageRelativePath/ImageFileName.jpg”
        - MarketingDesription: “description”
        - Products: null
        - ProductIds_0: 1
    - CategoryClassId: -1
    - CategoryClassName: “”
    - CategoryId: -1
    - Description: “”
    - ImageAltText: “”
    - ImageUrl: “”
    - MarketingDesription: “”
    - Products: null
- MenuId: 1
- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”

GetProductByProductId

This method is used to get product details by selected product identifier.

Input Parameters

Table 82 - GetProductByProductId

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetProductByProductId	A request object containing product identifier to get product details.

Output Parameters

Table 83 - GetProductByProductIdResponse

Type	Description
GetProductByProductIdResponse	A response object contains productDTO object consisting product details.

When to Use

This function can be called to get product details.

Example: Get Product by Product Id

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - ProductId: 1234

Response

- Product
    - Description: “OHEICS's newest taste sensation”
    - ImageAltText: “”
    - ImageUrl: “~/ImageRelativePath/ImageFileName.jpg”
    - MarketingDescription: “Description”
    - Price: 6.00
    - ProductId: 14
    - SortOrder: 0
    - …

GetProductByCategory

This method is used to get product details of type category by selected category and sub category identifiers.

Input Parameters

Table 84 - GetProductByCategoryRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetProductByCategoryRequest	A request object containing category and subcategory identifiers to get categorydetails.

Output Parameters

Table 85 - GetProductByCategoryResponse

Type	Description
GetProductByCategoryResponse	A response object contains CategoryDTO object consisting category details.

When to Use

This function can be called to get category details and list of products for category.

Example: Get Product by Category

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - CategoryId: 1234
    - SubCategoryId: 3453

Response

- Category
    - Categories: null
    - CategoryClassId: 4
    - CategoryClassName: “Class 1”
    - CategoryId: 5
    - Description: “Sides”
    - ImageAltText: “Image alt text descriptions”
    - ImageUrl: “~/ImageRelativePath/ImageFileName.jpg”
    - MarketingDesription: “description”

            - Products
                - Product_0
                    - Description: “OHEICS's newest taste sensation”
                    - ImageAltText: “”
                    - ImageUrl: “~/ImageRelativePath/ImageFileName.jpg”
                    - MarketingDescription: “Description”
                    - Price: 6.00
                    - ProductId: 14
                    - SortOrder: 0
- CategoryClassId: -1
- CategoryClassName: “”
- CategoryId: -1
- Description: “”
- ImageAltText: “”
- ImageUrl: “”
- MarketingDesription: “”
- Products: null

GetOrderOptions

This method is used to get the delivery option for the brand.

Input Parameters

Table 86 - GetOrderOptions

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

Output Parameters

Table 87 - GetOrderOptionsResponse

Type	Description
GetOrderOptionsResponse	A response object contains the list of OptionDTO object consisting order option details.

When to Use

This function can be called to get order options for populating the delivery option information.

Example: Get Order Option

Request

- Session
    - ApplicationId: “254c34ca-fe66-4871-b3d8-fd217f434405”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”

Response

- OrderOptionDTO_0
    - Description: Deliver to Front
    - OrderOptionId: 1
- OrderOptionDTO_1
    - Description: Deliver to Back
    - OrderOptionId: 2

GetBasketApplyDiscount

This method is to used apply a specific discount and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 88 - GetBasketApplyDiscountRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketApplyDiscountRequest	A request object containing the information required to be able to apply the discount. This inherits from the GetBasketRequest and therefore contains all of its properties.

Output Parameters

Table 89 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request. This inherits from the GetBasketResponse and therefore contains all of its properties.

When to Use

During the ordering process in a call center context, this method is called to apply a discount that has been granted to a customer as a resolution for a complaint.

Example

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - DiscountTypeId: 3
    - DiscountAmount: 10

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: “Product1”
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00
            - ProductId: 1
        - BasketItemDTO_1
            - BasketItemId: 2
            - Description: “Product2”
            - Price: 5.00
            - Quantity: 2
            - LineTotal: 10.00
            - ProductId: 2
    - NetItemTotal: 60.00
    - DeliveryChargeAmount: 5.00
    - DiscountAmount: 12:50
    - TaxChargeAmount: 9.00
    - BasketTotal: 61.50
    - AppliedDiscounts
        - [0]
            - BasketDiscountTypeId: 1

    - DiscountTypeId: 3
    - DiscountTypeText: “10% Discount”
    - IsDefaultDiscountAmountPercentage: true
    - DiscountAmount: 10
-
- BasketResponseStatus: Success
- IsSuccess: true
- IsVoucherApplied: false
-
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetBasketRemoveDiscount

This method is used to remove an existing discount from the basket and return the updated basket, including details managed and calculated by the basket services within OHEICS.

Input Parameters

Table 90 - GetBasketRemoveDiscountRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetBasketRemoveDiscountRequest	A request object containing the information required to be able to apply the discount. This inherits from the GetBasketRequest and therefore contains all of its properties.

Output Parameters

Table 91 - GetBasketResponse

Type	Description
GetBasketResponse	A response object containing the current basket and a response status providing details of any problems with the request. This inherits from the GetBasketResponse and therefore contains all of its properties.

When to Use

During the ordering process in a call center context, this method is called to remove a discount that has been previously applied to their current basket.

Example

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - DiscountTypeId: 3

Response

- Basket
    - BasketId: 1
    - BasketItems
        - BasketItemDTO_0
            - BasketItemId: 1
            - Description: “Product1”
            - Price: 10.00
            - Quantity: 5
            - LineTotal: 50.00
            - ProductId: 1
        - BasketItemDTO_1
            - BasketItemId: 2
            - Description: “Product2”
            - Price: 5.00
            - Quantity: 2
            - LineTotal: 10.00
            - ProductId: 2
    - NetItemTotal: 60.00
    - DeliveryChargeAmount: 5.00
    - DiscountAmount: 00.0
    - TaxChargeAmount: 9.00
    - BasketTotal: 61.50
    - AppliedDiscounts[]
-
- BasketResponseStatus: Success
- IsSuccess: true

- IsVoucherApplied: false
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

CancelOrder

This method will cancel an order in OHEICS.

Input Parameters

Table 92 - CancelOrderRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
CancelOrderRequest	A request object containing the information required to be able to cancel the order

Output Parameters

Table 93 - CancelOrderResponse

Type	Description
CancelOrderResponse	A response object containing the operation result. This inherits from the ServiceResponseBase, therefore contains all of its properties.

When to Use

Use this method to cancel an order that has already been placed. Not all of the orders can be cancelled; they need to comply with certain rules like being within the cancelation threshold. To find out if an order qualifies for cancelation, use GetOrderDetails method (inspect *CancellationAllowed* property in response) of CallOrderService.

Example

Request

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CancelOrderRequest
  - OrderId: 483902

        - CancellationComments: “Customer changed her mind and called back to cancel the order”

Response

- CancellOrderResponse
    - CancelOrderResult:Success
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

GetProductUpsellByMenuId

This method will get all product upsells of the menu.

Input Parameters

Table 94 - GetProductUpSellByMenuIdRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetProductUpSellByMenuIdRequest	A request object containing the information required to be able to filter the upsell.

Output Parameters

Table 95 - GetProductUpSellByMenuIdResponse

Type	Description
GetProductUpSellByMenuIdResponse	A response object containing the operation result. This inherits from the ServiceResponseBase, therefore contains all of its properties.

When to Use

When the frontend want to cache all the menu’s product upsell.

Example

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”

        - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- HidePastRequest
    - StoreId: 1
    - MenuId:1

Response

- GetProductUpSellByMenuIdResponse
    - ProductUpSellResponseStatus:Success
    - List<ProductUpSellDTO>:
        - UpSellId:1
        - EndDate: 12/31/9999
        - EndTime:12/31/9999 12:59:59
        - IsActive: true
        - IsAlwaysApplied: false
        - MinimumValue:0
        - OverrideMinimumValue:false
        - MinimumValue:0
        - Priority:999999
        - RuleDescription: Test Rule
        - RuleName: Product UpSell Test
        - StartDate:4/1/2014
        - StartTime:4/1/2014 12:00:00
        - ProductUpSellOffer:
            - OfferId:9
            - AddProductPermanently: false
            - DiscountAmount:10
            - IsCustomerAttributeAnded:false
            - IsFixedAmount:true
            - OfferKeyValuePairList:null
            - OverrideAmount:0
            - PriceOverride:false
            - UseAllProducts:true
            - OfferGroups:
                - GroupId:12
                - GroupName: Group 2
                - Quantity:1
                - Items:
                    - AddProductPermanently:false
                    - ProductId:1234
                    - Title:Coco 250ML

            - Description:Coco 250 ML
            - OfferPrice:9.9
            - PreOfferPrice:19.9
    - QualifyingGroups:
        - GroupId:5
        - GroupName: Group 5
        - Quantity:1
        - Items:
            - ProductId:8

Title: Chicken Bite 6PK

GetProductUpsellByProductId

This method will get all product upsells of the product.

Input Parameters

Table 96 - GetProductUpSellByProductIdRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetProductUpSellByProductIdRequest	A request object containing the information required to be able to filter the upsell

Output Parameters

Table 97 - GetProductUpSellByProductIdResponse

Type	Description
GetProductUpSellByProductIdResponse	A response object containing the operation result. This inherits from the ServiceResponseBase, therefore contains all of its properties.

When to Use

When the frontend want to cache all the menu's product upsell.

Example

Request

- Session

- ApplicationId: "OHEICS-123"
- CultureCode: "en-GB"
- SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"

- Request
  - StoreId: 1
  - Product:1

Response

- GetProductUpSellByMenuIdResponse
  - ProductUpSellResponseStatus:Success
  - List<ProductUpSellDTO>:
    - UpSellId:1
    - EndDate: 12/31/9999
    - EndTime:12/31/9999 12:59:59
    - IsActive: true
    - IsAlwaysApplied: false
    - MinimumValue:0
    - OverrideMinimumValue:false
    - MinimumValue:0
    - Priority:999999
    - RuleDescription: Test Rule
    - RuleName: Product UpSell Test
    - StartDate: 4/1/2014
    - StartTime: 4/1/2014 12:00:00
    - ProductUpSellOffer:
      - OfferId: 9
      - AddProductPermanently: false
      - DiscountAmount:10
      - IsCustomerAttributeAnded: false
      - IsFixedAmount: true
      - OfferKeyValuePairList: null
      - OverrideAmount: 0
      - PriceOverride: false
      - UseAllProducts:t rue
      - OfferGroups:
        - GroupId: 12
        - GroupName: Group 2
        - Quantity: 1
        - Items:
          - AddProductPermanently: false

        - ProductId: 1234
        - Title: Coco 250ML
        - Description: Coco 250 ML
        - OfferPrice: 9.9
        - PreOfferPrice: 19.9
- QualifyingGroups:
    - GroupId:5
    - GroupName: Group 5
    - Quantity: 1
    - Items:
        - ProductId: 8
        - Title: Chicken Bite 6PK

DeclineCurrentProductUpsell

This method will remove the current offer and reset the qualifier product as a normal product in the basket.

Input Parameters

Table 98 - DeclineCurrentProductUpsellRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
DeclineCurrentProductUpsellRequest	A request object containing the information of invitee

Output Parameters

Table 99 - DeclineCurrentProductUpsellResponse

Type	Description
DeclineCurrentProductUpsellResponse	A response object containing the operation result. This inherits from the ServiceResponseBase, therefore contains all of its properties.

When to Use

When the frontend want to decline the current product upsell.

Example

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - InviteeId: null
    - InviteeName: null

Response

- DeclineCurrentProductUpsellResponse
    - IsSuccess: true

Payment Service

The OHEICS payment service is used to process payment for orders. This service will be run over a secure encrypted connection (ssl/https).

GetCheckoutPaymentOptions

This method returns available payment methods, payment card options, and saved customer cards.

Payment methods include items such as cash, credit/debit cards.

Payment card options includes the card type and card title. The card type refers to the particular card type differentiating between debit, credit, and loyalty cards. The card title refers to the name of the payment network provider e.g. Visa or MasterCard. The PaymentCardOptionId identifies the payment option and will be used in the subsequent request to place the order.

Input Parameters

Table 100 - GetCheckoutPaymentOptionsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetCheckoutPaymentOptionsRequest	This request object extends ServiceRequestBase and doesn’t add any additional information at this moment in time

Output Parameters

Table 101 - GetCheckoutPaymentOptionsResponse

Type	Description
GetCheckoutPaymentOptionsResponse	This response object contains an array of payment methods available, an array of payment card options, and an array of any cards stored by the customer.

When to Use

This method should be used by clients to determine the payment options available to a customer in order to complete the order. The customer does not need to be registered and logged in to use this functionality.

Example: getcheckoutpaymentoptions called

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request

Response

- PaymentMethodsAvailable
    - Cash
    - CreditDebitCard
- PaymentCardOptions
    - PaymentCardOptionId: 1
    - CardType: Credit
    - CardTitle: Visa
    - PaymentCardOptionId: 2
    - CardType: Debit
    - CardTitle: Visa
- CustomerStoredCards
    - CustomerPaymentCardId: 1
    - CardNumberDisplayText: VISA **** **** **** 1111
    - CardType: Credit
    - PaymentCardOptionId: 1

PlaceOrder

A call to the PlaceOrder() method is made in order to process payment for the order. The customer can save the credit card details into their account. When a customer wants to place an order with the saved credit card use GetCustomerAccount to populate the available cards from their account and pass the id when the customer places an order.

Input Parameters

Table 102 - PlaceOrderRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
PlaceOrderRequest	The PlaceOrder request object contains delivery and billing address details, customer details (for unregistered customer orders) and an array of payments (so single or split payments can be processed).

Output Parameters

Table 103 - PlaceOrderResponse

Type	Description
PlaceOrderResponse	Contains the status of the order e.g. Success, InvalidDeliveryAddress.

When to Use

The PlaceOrder() method should be used by clients to complete and process payment for an order.

Example: Process Payment for Unregistered Customer – Collection, Cash Order

- placeorder called

Request#1

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
  - Title: Mr

    - FirstName: John
    - LastName: Smith
    - ContactTelephonePrimary: 0112345678
    - ContactEmailPrimary: john.smith@email.com
    - OrderPayments
        - Amount: 2.99

Response

- OrderStatus
    - Success
- OrderPromiseTime: 2010-01-14T16:00:00

Request#2

Placing an order with saved card

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - Title: Mr
    - FirstName: John
    - LastName: Smith
    - ContactTelephonePrimary: 0112345678
    - ContactEmailPrimary: john.smith@email.com
    - OrderPayments (PaymentStoredCardDTO)
        - Amount: 2.99
        - CardType: Credit
        - CustomerPaymentCardID: 1
        - SecurityCode:123

Request#3

Placing an order with credit card and saving the card into the customer card list

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - Title: Mr
    - FirstName: John
    - LastName: Smith
    - ContactTelephonePrimary: 0112345678
    - ContactEmailPrimary: john.smith@email.com

- SavePaymentCard: true
- OrderPayments (PaymentCardDTO)
  - Amount: 2.99
  - CardType: Credit
  - CardNumber: 4444333322221111
  - ExpireMonth: 12
  - ExpireYear:2013
  - SecurityCode: 123
  - SortCode: “00-00-00”
  - Name: John Smith
  - PaymentCardOptionId: 1

Request#4

Placing an order with cash. But it includes the channelpaymentmethodid, which is mapped to different tendermediaid (student card). So, when it goes to POS, it places an order with student card.

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
  - Title: Mr
  - FirstName: John
  - LastName: Smith
  - ContactTelephonePrimary: 0112345678
  - ChannelPaymentMethodId: 3
  - ContactEmailPrimary: john.smith@email.com
  - OrderPayments
    - Amount: 2.99

CalculateTotal

This method is used to calculate the tax of the current basket.

Input Parameters

Table 104 - PlaceOrderRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

PlaceOrderRequest	The PlaceOrder request object contains customer details (for unregistered customer orders) and an array of payments (so single or split payments can be processed) to calculate the tax.

Output Parameters

Table 105 - CalculateTotalResponse

Type	Description
CalculateTotalResponse	A response object containing information about the success or failure of the order including the tax and the order total.

When to Use

The CalculateTotal() method should be used by clients to calculate the tax of the items in the basket.

Example: Get Tax for the Current Basket Items

- Calculatetotal called

Request

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
  - Title: Mr
  - FirstName: John
  - LastName: Smith
  - ContactTelephonePrimary: 0112345678
  - ContactEmailPrimary: john.smith@email.com
  - OrderPayments
    - Amount: 2.99

Response

- TotalAmountOfTax: 1.19
- TotalPrice: 11.19
- TotalPriceOriginal: 10
- TotalAmountOfDiscount: 0
- TotalAmountOfSurcharge: 0
- ResultType: Success

- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetPaymentOptions

This method returns available payment methods, payment card options, and saved customer cards.

Payment methods include items such as cash, credit/debit cards. If there are multiple entries for the same paymenttype, this method returns the payment type along with the description and the channelpaymentmethodid. If there are more than one entry for the same payment method, place order method should include the channelpaymentmethodid while placing an order.

Payment card options includes the card type and card title. The card type refers to the particular card type differentiating between debit, credit and loyalty cards. The card title refers to the name of the payment network provider e.g. Visa or MasterCard. The PaymentCardOptionId identifies the payment option and will be used in the subsequent request to place the order.

Input Parameters

Table 106 - GetPaymentOptionsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetPaymentOptionsRequest	This request object extends ServiceRequestBase and doesn’t add any additional information at this moment in time

Output Parameters

Table 107 - GetPaymentOptionsResponse

Type	Description
GetPaymentOptionsResponse	This response object contains an array of payment methods available, an array of payment card options, and an array of any cards stored by the customer.

When to Use

The GetCheckoutPaymentOptions() method should be used by clients to determine the payment options available to a customer in order to complete the order. The customer does not need to be registered and logged in to use this functionality.

Example

- getcheckoutpaymentoptions called

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request

Response

- PaymentMethodsAvailable#1
    - ChannelPaymentMethodId: 1
    - MethodType: Cash
    - Method Description: Pay In Store
- PaymentMethodsAvailable#2
    - ChannelPaymentMethodId: 2
    - MethodType: CreditDebitCard
    - Method Description: CreditDebitCard
- PaymentMethodsAvailable#3
    - ChannelPaymentMethodId: 3
    - MethodType: Cash
    - Method Description: Pay By Student Id
- PaymentCardOptions
    - PaymentCardOptionId: 1
    - CardType: Credit
    - CardTitle: Visa
    - PaymentCardOptionId: 2
    - CardType: Debit
    - CardTitle: Visa
- CustomerStoredCards
    - CustomerPaymentCardId: 1
    - CardNumberDisplayText: VISA **** **** **** 1111
    - CardType: Credit
    - PaymentCardOptionId: 1

Customer Service

The OHEICS customer service is responsible for customer centric functionality such as login/logout and customer account management.

Login

This method is used to log in an existing registered user.

Input Parameters

Table 108 - LoginRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
LoginRequest	A request object containing the information required to be able to log the registered user in.

Output Parameters

Table 109 - LoginResponse

Type	Description
LoginResponse	A response object containing information about the successful or failed login attempt.

When to Use

The Login() method should be used when an existing customer is attempting to log into the system. If successful, the login service method will set appropriate information in the session which will enabled future service method calls by the calling application to be processed without the need for further secure information (apart from the session identifier passed as part of the session parameter).

The Login() method will be called over a secure channel, preventing discovery or abuse of user information, such as passwords.

Example: Existing Registered User John Smith Logs into the System

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
    - UserName: "john.smith@OHEICS.co.uk"

        - Password: “ThisIsMyPassword123”
        - CreatePersistentCookie:true

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

AddCustomerPaymentCard

This method is used to add customer payment card information.

Input Parameters

Table 110 - AddCustomerPaymentCardRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
AddCustomerPaymentCardRequest	A request object containing the payment card information to add into the customer’s account.

Output Parameters

Table 111 - AddCustomerPaymentCardResponse

Type	Description
AddCustomerPaymentCardResponse	A response object containing information of the payment card along with its unique identifier.

When to Use

The AddCustomerPaymentCard() method is used to add customer’s payment card information outside of placeorder request. We can add the paymentcard information while placing the order as well.

Example: Add Customer Payment Card

Request

- Session
    - ApplicationId: “OHEICS-123”

        - CultureCode: “en-GB”
        - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - CustomerPaymentCard (PaymentCardDTO):
        - CardNumber: 4444333322221111
        - CardType: Credit
        - ExpireMonth: 12
        - ExpireYear: 15
        - PaymentCardOptionId: 1
        - PaymentMethodType: CreditDebitCard
        - Name: Personal Chase Card

Response

- Response
    - CustomerPaymentCard (PaymentStoredCardDTO):
        - CardNumberDisplayText: VISA **** **** **** 1111
        - CustomerPaymentCardId: 1
        - PaymentCardOptionId: 1
    - ResultType: Success
    - IsSuccess: true
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

RemoveCustomerPaymentCard

This method is used to remove customer’s payment card information.

Input Parameters

Table 112 - RemoveCustomerPaymentCardRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
RemoveCustomerPaymentCardRequest	A request object containing the payment card identifier to remove from customer’s account.

Output Parameters

Table 113 - ServiceResponseBase

Type	Description
ServiceResponseBase	A response object containing information validation and flag if request has been successfully executed or not.

When to Use

The RemoveCustomerPaymentCard() method is used to remove customer's payment card information.

Example: Remove Customer Payment Card

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
    - CustomerPaymentCardId: 2344

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: ""
- Context: ""
- ValidationErrors: None

Register

The Register() service method is used to register a new user with the system, allowing them to enhance their ordering experience. A registered user can also store their details securely, including payment card details, address information, wishlist items, and previous order information.

Input Parameters

Table 114 - RegisterRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
RegisterRequest	A request object containing the information required to be able to register a new user with the system.

Output Parameters

Table 115 - RegisterResponse

Type	Description
RegisterResponse	A response object containing information about the success or failure of the registration request.

When to Use

The Register() service method should be used when a new user wishes to register their details with the system in order that future experience can be enhanced.

Examples: New User John Smith Registers with the System

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - Customer
        - CustomerId: 0
        - ContactTelephonePrimary: “01234567890”
        - ContactTelephoneSecondary: “”
        - ContactMobile: “07000123456”
        - Title: Title.Mr
        - FirstName: “John”
        - LastName: “Smith”
        - ContactEmailPrimary: “john.smith@OHEICS.co.uk”
        - ContactEmailSecondary: “”

    - MarketingOptIn: false
    - IsBlacklisted: false
    - CustomerAttributeList: None
    - ClientIPAddress: 172.28.212.22
- CustomerAddress
    - AddressId: 0
    - OrganisationName: “”
    - BuildingNumber: 0
    - BuildingLetter: “”
    - BuildingName: “2 St Andrews Court”
    - StreetName: “Wellington Street”
    - District: “”
    - TownCity: “Thame”
    - PostCodeOrZip: “OX9 3WT”
    - Longitude: 0.000000
    - Latitude: 0.000000
    - CountryId: 1
    - CountryText: “United Kingdom”
- Password: “ThisIsMyPassword123”

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

PasswordRecovery

The PasswordRecovery() method is used to request an email response to a customer who may have forgotten the password they provided for their registered logon to the system. When called, the method will generate and send an email to the registered customers primary contact email address with their logon details.

Input Parameters

Table 116 - PasswordRecoveryRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
PasswordRecoveryRequest	A request object containing the information required to be able to send a logon details confirmation email to the registered customer.

Output Parameters

Table 117 - PasswordRecoveryResponse

Type	Description
PasswordRecoveryResponse	A response object containing details as to the success or failure of the request.

When to Use

The PasswordRecovery() method should be used when a user requests that information about their logon credentials be emailed to their registered primary contact email address if they have forgotten their registered password.

Example: Existing Registered User John Smith Requests a Password Recovery Email

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - UserName: “john.smith@OHEICS.co.uk”

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetCustomerAccount

The GetCustomerAccount() service method returns details of the currently logged in customer or for the customer id in request (useful in a callcentre context), optionally including registered addresses and stored payment cards. Note that stored payment cards are only provided in shortened form and do not provide secure information which could be used in fraudulent behavior.

Input Parameters

Table 118 - GetCustomerAccountRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetCustomerAccountRequest	A request object containing the information required to be able to retrieve the customer account details.

Output Parameters

Table 119 - GetCustomerAccountResponse

Type	Description
GetCustomerAccountResponse	A response object containing customer account details and information as to the success or failure of the request.

When to Use

The GetCustomerAccount() service method should be used during the administration of a customer account, to retrieve and utilize customer details, registered address details and stored payment card details. This allows the end user to manage their own data within the system.

Note that a user must be logged in to use this service method. The details retrieved are for the current logged in user only.

Example: Get Customer Details for Registered Customer John Smith, Including Addresses

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - GetCustomerAddressDetails: true

    - GetCustomerPaymentCardDetails: true

Response

- CustomerDetails
    - CustomerId: 0
    - ContactTelephonePrimary: "01234567890"
    - ContactTelephoneSecondary: ""
    - ContactMobile: "07000123456"
    - Title: Title.Mr
    - FirstName: "John"
    - LastName: "Smith"
    - ContactEmailPrimary: "john.smith@OHEICS.co.uk"
    - ContactEmailSecondary: ""
    - MarketingOptIn: false
    - IsBlacklisted: false
    - CustomerAttributeList: None
- CustomerAddresses
    - AddressDTO_0
        - AddressId: 1
        - OrganisationName: ""
        - BuildingNumber: 0
        - BuildingLetter: ""
        - BuildingName: "2 St Andrews Court"
        - StreetName: "Wellington Street"
        - District: ""
        - TownCity: "Thame"
        - PostCodeOrZip: "OX9 3WT"
        - Longitude: 0.000000
        - Latitude: 0.000000
        - CountryId: 1
        - CountryText: "United Kingdom"
- CustomerPaymentCard#1:
    - CardType: Credit
    - CardNumber: CreditCardxxxxxxxxxxxx1111
    - CustomerPaymentCardId: 1
    - PaymentCardOptionId: 1
- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: ""

- Context: “”
- ValidationErrors: None

UpdateCustomer

The UpdateCustomer() service method updates the current logged in customer details with those provided in the service request.

Input Parameters

Table 120 - UpdateCustomerRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
UpdateCustomerRequest	A request object containing the information required to be able to update the customer details.

Output Parameters

Table 121 - UpdateCustomerResponse

Type	Description
UpdateCustomerResponse	A response object containing the updated customer details and information as to the success or failure of the request.

When to Use

The UpdateCustomer() method should be used during administration of the currently logged in customer. The end user of the calling application will be able to administer their own details.

Example: Update the Secondary Contact Email for Customer John Smith

Request

- Session
  - ApplicationId: “OHEICS-123”
  - CultureCode: “en-GB”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
  - Customer
    - CustomerId: 1
    - ContactTelephonePrimary: “01234567890”
    - ContactTelephoneSecondary: “”

        - ContactMobile: “07000123456”
        - Title: Title.Mr
        - FirstName: “John”
        - LastName: “Smith”
        - ContactEmailPrimary: “john.smith@OHEICS.co.uk”
        - ContactEmailSecondary: “john.smith2@OHEICS.co.uk”
        - MarketingOptIn: false
        - IsBlacklisted: false
        - CustomerAttributeList: None
    - UpdateAsSecondPerson: false

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- Update the secondary contact email for customer john smith (IN a Callcentre context)

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - Customer
        - CustomerId: 1
        - ContactTelephonePrimary: “01234567890”
        - ContactTelephoneSecondary: “”
        - ContactMobile: “07000123456”
        - Title: Title.Mr
        - FirstName: “John”
        - LastName: “Smith”
        - ContactEmailPrimary: “john.smith@OHEICS.co.uk”
        - ContactEmailSecondary: “john.smith2@OHEICS.co.uk”
        - MarketingOptIn: false
        - IsBlacklisted: false
        - CustomerAttributeList: None
    - UpdateAsSecondPerson: true

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

UpdatePassword

The UpdatePassword() service method updates the current logged in user’s password with new password.

Input Parameters

Table 122 - UpdatePasswordRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
UpdatePasswordRequest	A request object containing the information about old and new password.

Output Parameters

Table 123 - UpdatePasswordResponse

Type	Description
UpdatePasswordResponse	A response object containing information about validation and flag indicating whether password has been updated successfully.

When to Use

The UpdatePassword() method should be used during administration of the currently logged in customer. The end user of the calling application will be able to administer their own details.

Example: Update the Secondary Contact Email for Customer John Smith

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”

- Request
    - Customer
        - CustomerId: 1
        - ContactTelephonePrimary: “01234567890”
        - ContactTelephoneSecondary: “”
        - ContactMobile: “07000123456”
        - Title: Title.Mr
        - FirstName: “John”
        - LastName: “Smith”
        - ContactEmailPrimary: “john.smith@OHEICS.co.uk”
        - ContactEmailSecondary: “john.smith2@OHEICS.co.uk”
        - MarketingOptIn: false
        - IsBlacklisted: false
        - CustomerAttributeList: None

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

AddAddress

The AddAddress() service method allows an additional address to be added to the currently logged in customer or to the customer id in request (in a call center context). Note that a registered user MUST have at least one registered address. The registration method cannot be called without an address, and the RemoveAddress method cannot be called to remove the only address stored against a registered customer.

Input Parameters

Table 124 - AddAddressRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
AddAddressRequest	A request object containing the information required to be able to add an address to the registered customer.

Output Parameters

Table 125 - AddAddressResponse

Type	Description
AddAddressResponse	A response object containing the address created and a status indicating the success or failure of the request.

When to Use

The AddAddress() method should be used during administration when the user can add additional addresses to their registration.

Example: Add a New Address to Registered Customer John Smith

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - CustomerAddress
        - AddressId: 0
        - OrganisationName: “”
        - BuildingNumber: 50
        - BuildingLetter: “”
        - BuildingName: “”
        - StreetName: “High Street”
        - District: “”
        - TownCity: “Thame”
        - PostCodeOrZip: “OX9 1FG”
        - Longitude: 0.000000
        - Latitude: 0.000000
        - CountryId: 1
        - CountryText: “United Kingdom”
    - CustomerId: null

Response

- CustomerAddress
    - AddressId: 2
    - OrganisationName: “”
    - BuildingNumber: 50

    - BuildingLetter: “”
    - BuildingName: “”
    - StreetName: “High Street”
    - District: “”
    - TownCity: “Thame”
    - PostCodeOrZip: “OX9 1FG”
    - Longitude: 0.000000
    - Latitude: 0.000000
    - CountryId: 1
    - CountryText: “United Kingdom”
- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- Add a new address to registered customer john smith (IN a Callcentre context)

Where Customer id for John Smith is: 48940

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - CustomerAddress
        - AddressId: 0
        - OrganisationName: “”
        - BuildingNumber: 50
        - BuildingLetter: “”
        - BuildingName: “”
        - StreetName: “High Street”
        - District: “”
        - TownCity: “Thame”
        - PostCodeOrZip: “OX9 1FG”
        - Longitude: 0.000000
        - Latitude: 0.000000
        - CountryId: 1
        - CountryText: “United Kingdom”
    - CustomerId: 48940

Response

- CustomerAddress
    - AddressId: 2
    - OrganisationName: “”
    - BuildingNumber: 50
    - BuildingLetter: “”
    - BuildingName: “”
    - StreetName: “High Street”
    - District: “”
    - TownCity: “Thame”
    - PostCodeOrZip: “OX9 1FG”
    - Longitude: 0.000000
    - Latitude: 0.000000
    - CountryId: 1
    - CountryText: “United Kingdom”
- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

RemoveAddress

The RemoveAddress() service method removes an existing address for the currently logged in customer. Note that a registered customer must have at least one address. Calling the RemoveAddress() method to remove the only address for a registered user will return a failure status.

Input Parameters

Table 126 - RemoveAddressRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
RemoveAddressRequest	A request object containing the information required to be able to remove the registered customer address.

Output Parameters

Table 127 - RemoveAddressResponse

Type	Description
RemoveAddressResponse	A response object containing a status value indicating the success or failure of the request.

When to Use

The RemoveAddress() method should be used during administration to remove an existing address for the current registered customer. As mentioned above, note that a registered customer must have at least one address. Attempted removal of the only address for a registered user will result in a failure response status.

Example: Remove the Address with Id 2 from Registered Customer John Smith

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - CustomerAddressId: 2
    - CustomerId: null

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- Remove the Address with Id 2 from registered customer john smith (IN a Callcentre CONTEXT)

Where Customer id for John Smith is: 48940

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - CustomerAddressId: 2
    - CustomerId: 48940

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

UpdateAddress

The UpdateAddress() service method updates an existing address for the currently logged in user or the customer id in request (in a call center context).

Input Parameters

Table 128 - UpdateAddressRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
UpdateAddressRequest	A request object containing the information required to be able to update the existing customer address.

Output Parameters

Table 129 - UpdateAddressResponse

Type	Description
UpdateAddressResponse	A response object containing details of the updated address and a status to indicate the success or failure of the request.

When to Use

The UpdateAddress() method should be used during administration of the current logged in customer details in order that the end user can manage their own address details.

Example: Logged in User John Smith Updates an Address with a New District

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”

- Request
    - CustomerAddress
        - AddressId: 1
        - OrganisationName: “”
        - BuildingNumber: 0
        - BuildingLetter: “”
        - BuildingName: “2 St Andrews Court”
        - StreetName: “Wellington Street”
        - District: “Oxfordshire”
        - TownCity: “Thame”
        - PostCodeOrZip: “OX9 3WT”
        - Longitude: 0.000000
        - Latitude: 0.000000
        - CountryId: 1
        - CountryText: “United Kingdom”
    - CustomerId: null

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None
- Updates john smith’s address with a new district (In a Callcentre context)

Where Customer id for John Smith is: 48940

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - CustomerAddress
        - AddressId: 1
        - OrganisationName: “”
        - BuildingNumber: 0
        - BuildingLetter: “”
        - BuildingName: “2 St Andrews Court”
        - StreetName: “Wellington Street”
        - District: “Oxfordshire”

        - TownCity: “Thame”
        - PostCodeOrZip: “OX9 3WT”
        - Longitude: 0.000000
        - Latitude: 0.000000
        - CountryId: 1
        - CountryText: “United Kingdom”
    - CustomerId: 48940

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

GetCustomerOrders

The GetCustomerOrders() method returns recent orders and wish lists (favorites) for a customer.

Input Parameters

Table 130 - GetCustomerOrdersRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetCustomerOrdersRequest	The request object.

Output Parameters

Table 131 - GetCustomerOrdersResponse

Type	Description
GetCustomerOrdersResponse	A response object containing an array of WishList objects and an array of RecentOrder objects.

When to Use

The GetCustomerOrders () method should be used during the order process to retrieve key data relating to previous orders / saved basket items (WishLists) in order to re-create new orders based on these items.

Example: Logged in User John Smith Retrieves Customer Orders

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetCustomerOrdersRequest

Response

- WishList#1
    - BasketId:1
    - BasketDescription: “Favorite Order 1”
    - ChannelId: 1
    - Channel Text: “A Channel”
    - ExpiryDate: “2009-12-31”
    - ExpiryDateAsString: “31/12/2009 10:58:31”
    - MenuId: 1
    - SavedDate: “2009-12-30”
    - StoreId: 1
    - StoreName: “A Store”
    - Items
- Description
- Quantity
- ProductId
- DisplayPrice
- Items
-
- WishList#2
    - BasketId:2
    - BasketDescription: “Favorite Order 2”
    - ChannelId: 1
    - Channel Text: “A Channel”
    - ExpiryDate: “2009-12-31”
    - ExpiryDateAsString: “31/12/2009 10:58:31”
    - MenuId: 1
    - SavedDate: “2009-12-30”
    - StoreId: 1
    - StoreName: “B Store”
    - Items
- Description
- Quantity

- ProductId
- DisplayPrice
- Items
- RecentOrder#1
    - OrderDate: “2009-12-31”
    - OrderId: 1
    - OrderRefernce
    - PosConfirmationNumber
    - Moniker
    - StoreId: 1
    - StoreName: “A Store”
    - OrderClass
    - TotalOrderValue: 12.99
    -
    - Items
        - ProductId
        - ProductText
        - ProductDescription
        - Quantity
        - DisplayPrice
        - IsMainproduct
        - Items
- RecentOrder#2
    - Orderdate: “2009-12-31”
    - OrderId: 2
    - OrderRefernce
    - PosConfirmationNumber
    - Moniker
    - StoreId: 2
    - StoreName: “B Store”
    - TotalOrderValue: 10.99
    - Items
        - ProductId
        - ProductText
        - ProductDescription
        - Quantity
        - DisplayPrice
        - IsMainproduct
        - Items

- TotalRecentOrderCount
- AllowOrderPlacement

GetCustomerFavorite

The GetCustomerFavorite() method returns favorites for a customer.

Input Parameters

Table 132 - GetCustomerFavoritesRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetCustomerFavoritesRequest	The request object.

Output Parameters

Table 133 - GetCustomerFavoritesResponse

Type	Description
GetCustomerFavoritesResponse	A response object containing an array of favorite objects.

When to Use

The GetCustomerFavorite() method should be used to retrieve key data relating to basket or basket items in order to re-create new orders based on these items.

Example: Logged in User John Smith Retrieves Customer Orders

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetCustomerFavoritesRequest

Response

- Favorite#1
    - Favorite Id:1
    - Favorite Name: “Favorite 1”
    - Client Id: 1
    - Client Text: “A Client”
    - User Account: 1

    - Brand Id: 1
    - Brand Text: “Brand Name”
    - Is Group : true
    - Items
- Description
- Quantity
- ProductId
- DisplayPrice
- Items

RemoveFavorite

The RemoveFavorite() method will mark the favorite for deletion but will not delete it.

Input Parameters

Table 134 - RemoveFavoriteRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
RemoveFavoriteRequest	The request object.

Output Parameters

Table 135 - RemoveOrderResponse

Type	Description
RemoveOrderResponse	A response object containing ResponseStatus of deleting favorite.

When to Use

The RemoveFavorite() method should be used to mark a favorite as deleted.

Example: Logged in User John Smith Retrieves Customer Orders

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- RemoveFavoriteRequest

Response

- ResponseStatus
    - Success

SaveAsFavorite

The SaveFavorite() method will save whole basket or basket items to existing favorite or new favorite

Input Parameters

Table 136 - SaveAsFavoriteRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
SaveAsFavoriteRequest	The request object.

Output Parameters

Table 137 - SaveAsFavoriteResponse

Type	Description
SaveAsFavoriteResponse	A response object containing status of saving favorite.

When to Use

The SaveAsFavorite() method should be used to save a whole basket or basket items into new or existed favorite.

Example: Logged in User John Smith Retrieves Customer Orders

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- SaveAsFavoriteRequest

Response

- IsSuccess: true

CheckProductAvailability

The CheckProductAvailability() method will check the available of product id list and return the unavailable product id.

Input Parameters

Table 138 - CheckProductAvailabilityRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
CheckProductAvailabilityRequest	The request object.

Output Parameters

Table 139 - CheckProductAvailabilityResponse

Type	Description
CheckProductAvailabilityResponse	A response object containing list of product id that unavailable.

When to Use

The CheckProductAvailability() method will check the available of product id

Example: Logged in User John Smith Retrieves Customer Orders

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CheckProductAvailabilityRequest

Response

- NotAvailableProductIds
    - 12

UserAccountStatus

The UserAccountStatus() method returns the status of given user's account. This function will check whether user's account is active or locked.

Input Parameters

Table 140 - UserAccountStatusRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
UserAccountStatusRequest	The request object.

Output Parameters

Table 141 - UserAccountStatusResponse

Type	Description
UserAccountStatusResponse	A response object containing status of user's account.

When to Use

The UserAccountStatus () method should be used to check specific user's account status.

Example: User Account Status for User "Test"

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
    - UserName: "Test"

Response

- ResultType: IsActive

ForgotPassword

The ForgotPassword() method returns if user is allowed to recover forgot password and if user is allowed to recover forgot password it sends an email to user with reset password link.

Input Parameters

Table 142 - ForgotPasswordRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
ForgotPasswordRequest	The request object containing user and base urls and its expiry information.

Output Parameters

Table 143 - ForgotPasswordResponse

Type	Description
ForgotPasswordResponse	A response object containing status of forgot password request if user is allowed to recover forgot password.

When to Use

The ForgotPassword() method should be used to generate reset password links when a user wants to recover a forgotten password.

Example: Forgot Password for User "Test"

Request

- Session
  - ApplicationId: "OHEICS-123"
  - CultureCode: "en-GB"
  - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
  - ConfirmBaseUrl: https://OHEICS/secure/forgotpassword
  - LinkExpireAfterThisTime:15.30
  - UserName: "Test"

Response

- ResponseSatus: Success

ResetPassword

The ResetPassword() method reset password for user and unlock account if it is locked. Once password is reset user is notified by email for password reset confirmation.

Input Parameters

Table 144 - ResetPasswordRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
ResetPasswordRequest	The request object containing user and base urls and its expiry information and new password information.

Output Parameters

Table 145 - ResetPasswordResponse

Type	Description
ResetPasswordResponse	A response object containing status of reset password request and email where it sent reset password confirmation email.

When to Use

The ResetPassword () method should be used to reset password information and unlock the account.

Example: Reset Password for User “Test”

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- Request
    - Password: “Password1”
    - RetypedPassword: “Password1”
    - url: https://OHEICS/secure/ResetPassword
    - ResetRequestedId: {15269C6A-78FA-4F01-8793-279650AB973B}

Response

- ResponseSatus: Success

- UserEmailAddress: user@OHEICS.co.uk

IsPasswordResetRequestIdValid

The IsResetPasswordRequestIdValid() method validates the reset password request token and returns a Boolean flag.

Input Parameters

Table 146 - Guid

Type	Description
Guid	Reset password request id token

Output Parameters

Table 147 - Bool

Type	Description
Bool	Flag indicating whether reset password request token is valid.

When to Use

The IsResetPasswordRequestIdValid () method should be used to validate reset password request token before we send request for reset password.

Example: Is Password Reset Request Id Valid with 154c74ca-fe66-4871-b3d8-fd217f434402

Request

- resetRequestId:154c74ca-fe66-4871-b3d8-fd217f434402

Response

- true

GetFacebookFriendsPreviousOrders

The GetFacebookFriendsPreviousOrders() method gets a list of orders for customer's Facebook friends.

Input Parameters

Table 148 - List<long>

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.

List<long>	The list of the customer's Facebook friends' user identifiers.

Output Parameters

Table 149 - GetCustomerOrdersResponse

Type	Description
GetCustomerOrdersResponse	A response object containing an array of WishList objects and an array of RecentOrder objects.

When to Use

The GetFacebookFriendsPreviousOrders () method should be used to get a list of Facebook friends' orders.

Example: Get Facebook Friends' Previous Orders

Request

- Session
    - ApplicationId: "OHEICS-123"
    - CultureCode: "en-GB"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- facebookFriendsUserIds
    - long_0
        - "15269C6A-78FA-4F01-8793-279650AB973B"
        - "279650AB-4F01-4F01-78FA -589656ZB973C"

Response

- WishList#1
    - BasketId:1
    - BasketDescription: "Favorite Order 1"
    - ChannelId: 1
    - Channel Text: "A Channel"
    - ExpiryDate: "2009-12-31"
    - ExpiryDateAsString: "31/12/2009 10:58:31"
    - MenuId: 1
    - MenuText: "A Menu"
    - OrderTypeId: 1
    - OrderTypeText: "Collection"
    - SavedDate: "2009-12-30"
    - SavedDateAsString: "30/12/2009 23:00:00"
    - StoreId: 1

  - StoreText: “A Store”
- WishList#2
  - BasketId:2
  - BasketDescription: “Favorite Order 2”
  - ChannelId: 1
  - Channel Text: “A Channel”
  - ExpiryDate: “2009-12-31”
  - ExpiryDateAsString: “31/12/2009 10:58:31”
  - MenuId: 1
  - MenuText: “B Menu”
  - OrderTypeId: 2
  - OrderTypeText: “Delivery”
  - SavedDate: “2009-12-30”
  - SavedDateAsString: “30/12/2009 23:00:00”
  - StoreId: 1
  - StoreText: “B Store”
- RecentOrder#1
  - ChannelId
  - ChannelText: “A Channel”
  - DateOrderRequired: “2009-12-31”
  - DateOrderRequiredAsString: “31/12/2009 10:58:31”
  - MenuId: 1
  - MenuText: “A Menu”
  - OrderId: 1
  - OrderTypeId: 1
  - OrderTypeText: “Collection”
  - StoreId: 1
  - StoreText: “A Store”
  - TotalOrderValue: 12.99
  - TotalOrderValueAsString: “$12.99”
- RecentOrder#2
  - ChannelId
  - ChannelText: “A Channel”
  - DateOrderRequired: “2009-12-31”
  - DateOrderRequiredAsString: “31/12/2009 10:58:31”
  - MenuId: 2
  - MenuText: “B Menu”
  - OrderId: 1
  - OrderTypeId: 2

- OrderTypeText: “Delivery”
- StoreId: 2
- StoreText: “B Store”
- TotalOrderValue: 10.99
- TotalOrderValueAsString: “$10.99”

AddFacebookUserOrder

The AddFacebookUserOrder() method is used to add customer’s Facebook user identifier to the order.

Input Parameters

Table 150 - AddFacebookUserOrder

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
Int	Customer Identifier
Long	Facebook User identifier
Int	Order identifier

Output Parameters

Table 151 - ServiceResponseBase

Type	Description
ServiceResponseBase	A response object containing information validation and flag if request has been successfully executed or not.

When to Use

After order confirmation, AddFacebookUserOrder() can be called to link a customer’s Facebook userid with latest order.

Example: Add Facebook User Order

Request

- Session
    - ApplicationId: “OHEICS-123”
    - CultureCode: “en-GB”

- SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- Request
  - CustomerId: 2344
  - facebookUserId:6568965656
  - OrderId: 4578

Response

- ResultType: Success
- IsSuccess: true
- ErrorType: None
- ErrorMessage: ""
- Context: ""
- ValidationErrors: None

GetCustomerNotes

This method will return the existing customer notes items in the OHEICS system for the current customer in session.

Input Parameters

Table 152 - GetCustomerNotesRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetCustomerNotesRequest	A request object containing the information needed to find the customer notes and delimit the results (Inherits from GetCustomerNotesRequestBase).

Output Parameters

Table 153 - GetCustomerNotesResponse

Type	Description
GetCustomerNotesResponse	A response object containing the list of customer notes.

When to Use

Whenever the existing customer notes for a given customer need to be displayed in the frontend.

Example

Request

- Session
  - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
  - CultureCode: “en-US”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetCustomerNote request
  - CustomerId: null
  - NumberOfNotesToDisplay: 15
  - Option: ParentAndChild
  - RootParentId: null
  - OrderId: null
  - DaysToShow: 30
  - IsSimplifiedView: true

Response

- CustomerNotes
  - [0]
    - CustomerNoteId: 43849
    - CustomerId: 432
    - TypeId: 3
    - TypeText: General
    - ClassId: 2
    - ClassText: General Note
    - ReasonCodeId: null
    - ReasonCodeText: null
    - ParentId: null
    - RootParentId: null
    - OrderId: null
    - NoteText: Customer always ask for low-carb food
    - IsProcessed: false
    - UpdatedByName: John Doe
    - OrderReference: null
    - DateUpdated: 11/05/2013
    - TimeZoneForDateCreated: "Mountain Standard Time"
    - ChildNotes:[]
-
  - IsSuccess: true
  - ErrorType: None
  - ErrorMessage: “”
  - Context: “”
  - ValidationErrors: None

GetCustomerComplaints

This method will return the existing customer complaint note items in the OHEICS system for the current customer in session.

Input Parameters

Table 154 - GetCustomerComplaintsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetCustomerComplaintsRequest	A request object containing the information needed to find the customer complaint notes and delimit the results (Inherits from GetCustomerNotesRequestBase).

Output Parameters

Table 155 - GetCustomerComplaintsResponse

Type	Description
GetCustomerComplaintsResponse	A response object containing the list of customer complaints notes.

When to Use

Use this method to retrieve the list of existing customer complaints for the customer in session.

Example: Retreive Parent Customer Complaints

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetCustomerNote request
    - CustomerId: null
    - NumberOfNotesToDisplay: null
    - Option: ParentOnly
    - RootParentId: null
    - OrderId: null
    - DaysToShow: null

        - IsSimplifiedView: false

Response

- CustomerNotes
    - [0]
        - CustomerNoteId: 4343
        - CustomerId: 6654
        - TypeId: 5
        - TypeText: “Complaint”
        - ClassId: 6
        - ClassText: “Customer Complaint”
        - ReasonCodeId: null
        - ReasonCodeText: null
        - ParentId: null
        - RootParentId: null
        - OrderId: null
        - NoteText: Order arrived one hour late
        - IsProcessed: false
        - UpdatedByName: John Doe
        - OrderReference: null
        - DateUpdated: 11/05/2013
        - TimeZoneForDateCreated: "Mountain Standard Time"
        - ChildNotes:[]

    - IsSuccess: true
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

Example: Retreive Child Customer Complaints (Resolutions)

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetCustomerNote request
    - CustomerId: null
    - NumberOfNotesToDisplay: null
    - Option: ChildOnly

    - RootParentId: 4343
    - OrderId: null
    - DaysToShow: null
    - IsSimplifiedView: false

Response

- CustomerNotes
    - [0]
        - CustomerNoteId: 786
        - CustomerId: 6654
        - TypeId: 7
        - TypeText: “Resolution”
        - ClassId: 9
        - ClassText: “Discount”
        - ReasonCodeId: 8
        - ReasonCodeText: “10% Discount”
        - ParentId: 4343
        - RootParentId: 4343
        - OrderId: null
        - NoteText: “10% Discount granted”
        - IsProcessed: true
        - UpdatedByName: John Doe
        - OrderReference: null
        - DateUpdated: 11/05/2013
        - TimeZoneForDateCreated: "Mountain Standard Time"
        - ChildNotes:[]

    - IsSuccess: true
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

GetCustomerNoteTypes

This method will return the customer note types currently defined in the OHEICS system.

Input Parameters

Table 156 - GetCustomerNoteTypesRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetCustomerNoteTypesRequest	A request object containing the information needed to find the customer note types.

Output Parameters

Table 157 - GetCustomerNoteTypesResponse

Type	Description
GetCustomerNoteTypesResponse	A response object containing the list of customer note types.

When to Use

Use this method to retrieve the list of customer note types (complaints, information, etc.).

Example

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetCustomerNote request
    - CustomerNoteTypeClass: CustomerComplaint
    - BrandId: null

Response

- CustomerNotesTypes
    - [0]
        - CustomerNoteTypeId: 1
        - CustomerNoteTypeClass: CustomerComplaint
        - DisplayTitleText: “Bad quality food”
    - [1]
        - CustomerNoteTypeId: 2
        - CustomerNoteTypeClass: CustomerComplaint
        - DisplayTitleText: “Food too expensive”

- IsSuccess: true
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

CreateCustomerNote

This method will create a customer note for the current customer in session.

Input Parameters

Table 158 - CreateCustomerNoteRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
CreateCustomerNoteRequest	A request object containing the information needed create the customer note.

Output Parameters

Table 159 - CreateCustomerNoteResponse

Type	Description
CreateCustomerNoteResponse	A response object containing the result of the operation and the identifier of the created customer note.

When to Use

This method can be used to create customer notes of different types, including *Information* notes, *Customer Complaint* notes, and *Resolution* notes.

Example: Creating a Customer Complaint

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CreateCustomerNote request
    - TypeId: 4
    - TypeText: “Customer Complaint”

    - ReasonCodeId: 6
    - ReasonCodeText: “Bad quality food”
    - ParentId: null
    - RootParentId: null
    - OrderId: null
    - NoteText: “The customer is complaining about the food quality”
    - IsProcessed: false
    - OrderReference: null

Response

- CreateCustomerNote Response
    - CustomerNoteId: 4578
    - IsSuccess: true
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

Example: Creating a Resolution Customer Note

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CreateCustomerNote request
    - TypeId: 6
    - TypeText: “Resolution”
    - ReasonCodeId: 6
    - ReasonCodeText: “10% Discount”
    - ParentId: 4578
    - RootParentId: 4578
    - OrderId: null
    - NoteText: “A 10% has been granted to the customer”
    - IsProcessed: true
    - OrderReference: null

Response

- CreateCustomerNote Response
    - CustomerNoteId: 4579
    - IsSuccess: true
    - ErrorType: None

    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

Example: Creating a General Customer Note (Information)

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CreateCustomerNote request
    - TypeId: 2
    - TypeText: “CustomerNote”
    - ReasonCodeId: 4
    - ReasonCodeText: “Other”
    - ParentId: null
    - RootParentId: null
    - OrderId: null
    - NoteText: “This customer always call in the mornings”
    - IsProcessed: false
    - OrderReference: null

Response

- CreateCustomerNote Response
    - CustomerNoteId: 4586
    - IsSuccess: true
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

CustomerSearch

This method will search for customers based on search parameters.

Input Parameters

Table 160 - CustomerSearchRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
CustomerSearchRequest	A request object containing the search parameters.

Output Parameters

Table 161 - CustomerSearchResult

Type	Description
CustomerSearchResult	A response object containing the list of matching customers.

When to Use

In a call center context, use this method to search for the customer who is making the call so that the take order process can start.

Example

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CustomerSearch request
    - LastName: null
    - PhoneNumber: null
    - StreetName: null
    - Email: “jdoe@mail.com”
    - TownCity: null
    - PostCodeOrZip: null
    - Status: NOT_SET

Response

- CustomerSearch Response
    - Customer
        - CustomerId: 8493

    - FirstName: "John"
    - LastName:"Doe"
    - Status:"Normal"
    - IsBlackListed:false
    - (*the rest of the CustomerDTO properties …*)
- CustomeAddress
    - AddressId: 4027
    - TownCity: "Columbia"
    - StreetName: "Columbia Gateway Dr."
    - TradeZoneId: 90303
    - (*the rest of the AddressDTO properties*)
- IsDiscountRequired: true
- Discount Reason: "10% discount has been granted"
- LastOrder: 11/05/2013
- ErrorType: None
- ErrorMessage: ""
- Context: ""
- ValidationErrors: None

SelectCustomerAndAddress

This method will set the given customer and address to the current session.

Input Parameters

Table 162 - SelectCustomerAndAddressRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
SelectCustomerAndAddressRequest	A request object containing the customer, address and tradezone to be set in session.

Output Parameters

Table 163 - SelectCustomerAndAddressResponse

Type	Description
SelectCustomerAndAddressResponse	A response object containing the result of the operation and the customer details (if flag in request is true).

When to Use

In a call center context, use this method to set the selected customer and address to the session so that the take order process can start. It is also used to retrieve call center-related information like pending discounts or complaints, etc.

Example

Request

- Session
    - ApplicationId: "A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA"
    - CultureCode: "en-US"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- SelectCustomerAndAddressRequest
    - CustomerId: 8493
    - AddressId: 4027
    - TradezoneId: 90303
    - ReturnCustomerDetails: true

Response

- SelectCustomerAndAddress Response
    - CustomerDetails
        - Customer
            - CustomerId: 8493
            - FirstName: "John"
            - LastName:"Doe"
            - Status:"Normal"
            - IsBlackListed:false
            - (*the rest of the CustomerDTO properties …*)
        - LastOrder
            - OrderType:Collection
            - Total: 545.54
            - (*the rest of the OrderDTO properties*)
        - LatestComplaint
            - TypeId:3
            - TypeText: "Complaint"
            - CustomerNoteId: 9039
            - *(the rest of the CustomerNoteDTO properties*)
    - AverageCustomerOrderValue: 46.98
    - ComplaintCount: 15
    - PendingDiscountIssuedOn: 11/05/2013
    - PendingDiscountIssuedBy: "Operator 1"

- PendingDiscountType: “Discount”
- PendingDiscountReason: “10% Discount”
- ErrorType: None
- ErrorMessage: “”
- Context: “”
- ValidationErrors: None

CreateCustomer

This method will create a new customer in the OHEICS system.

Input Parameters

Table 164 - CreateCustomerRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
CreateCustomerRequest	A request object containing the information required to create the customer.

Output Parameters

Table 165 - CreateCustomerResponse

Type	Description
CreateCustomerResponse	A response object containing the result of the operation and the customer address identifiers.

When to Use

In a call center context, to create new customers.

Example

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CreateCustomerRequest
    - Customer
        - FirstName: John

        - LastName:Doe
        - ContactTelephonePrimary: “5456647676”
        - (The rest of the *CustomerDTO* properties)
    - Address
        - TownCity: “Columbia”
        - StreetName: “Columbia Gateway Dr.”
        - BuildingNumber: “4”
        - (The rest of the *AddressDTO* properties)

Response

- CreateCustomer Response
    - CustomerId: 489302
    - CustomerAddressId: 399403
    - ErrorType: None
    - ErrorMessage: “”
    - Context: “”
    - ValidationErrors: None

AddCustomerPaymentCardWithoutCheckout

This method will add a specified card for a customer in the OHEICS system.

Input Parameters

Table 166 - AddCustomerPaymentCardRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
AddCustomerPaymentCardRequest	A request object containing the payment card of the customer.

Output Parameters

Table 167 - AddCustomerPaymentCardRequest

Type	Description
AddCustomerPaymentCardRequest	A request object containing the payment card of the customer.

When to Use

Add a payment card for a customer, but not required to go through checkout process.

Example

Request

- Session
  - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
  - CultureCode: “en-US”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- AddCustomerPaymentCardRequest
  - CustomerPayment
    - PaymentMethodType: LoyaltyCard
    - Name: “Lin Wang”
    - CardNumber: “4564710000000004”
    - Pin: “847”
    - ValidMonth: 2
    - ValidYear: 2014
    - ExpireMonth: 2
    - ExpireYear: 2019
    - CardType: Loyalty

Response

- AddCustomerPaymentCardResponse
  - CustomerPaymentCard
    - CustomerPaymentCardId: 100
    - CardNumberDisplayText: “Quantum **** **** 1111”
    - CardType: Loyalty
    - PaymentCardOptionId: 1
    - Pin: “847”

UpdateCustomerPaymentCardWithoutCheckout

This method will update a specified card for a customer in the OHEICS system.

Input Parameters

Table 168 - UpdateCustomerPaymentCardRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
UpdateCustomerPaymentCardRequest	A request object containing the payment card of the customer.

Output Parameters

Table 169 - UpdateCustomerPaymentCardResponse

Type	Description
UpdateCustomerPaymentCardResponse	A response object containing the updated customer payment card.

When to Use

Update a payment card for a customer, but not required to go through checkout process.

Example

Request

- Session
    - ApplicationId: "A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA"
    - CultureCode: "en-US"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- UpdateCustomerPaymentCardRequest
    - CustomerPayment
        - PaymentMethodType: LoyaltyCard
        - Name: "Lin Wang"
        - CardNumber: "4564710000000004"
        - Pin: "847"
        - ValidMonth: 2
        - ValidYear: 2014
        - ExpireMonth: 2
        - ExpireYear: 2019
        - CardType: Loyalty

Response

- AddCustomerPaymentCardResponse
    - CustomerPaymentCard
        - CustomerPaymentCardId: 100
        - CardNumberDisplayText: "Quantum **** **** 1111"
        - CardType: Loyalty
        - PaymentCardOptionId: 1
        - Pin: "847"

Check Service

The OHEICS check service is responsible for check interactions with POS.

CreateCheck

The CreateCheck() method is used to create open check in POS.

Input Parameters

Table 170 - OpenCheckRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
OpenCheckRequest	The open check request object.

Output Parameters

Table 171 - OpenCheckResponse

Type	Description
OpenCheckResponse	The open check response object.

When to Use

This method should be called to create open check in POS.

Example

Request

- Session
  - ApplicationId: "A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA"
  - CultureCode: "en-US"
  - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- OpenCheckRequest
  - Context (string)
  - CheckID (string)
  - TableNumber (string)
  - CoverCounts (int 32)
  - StoreId (string)

Response

- OpenCheckResponse
  - CkeckID (string) – yes, this is the spelling in the code!
  - CheckSequenceNumber (string)
  - CheckNumber (string)
  - CheckId (string)
  - CheckGuid (GUID)

PayCheck

The PayCheck() method used to pay amount for existing open check in POS.

Input Parameters

Table 172 - CheckPaymentRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
CheckPaymentRequest	The check payment request object.

Output Parameters

Table 173 - CheckPaymentResponse

Type	Description
CheckPaymentResponse	The check payment response object.

When to Use

This method should be called to make payment for existing open check in POS.

Example

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- CheckPaymentRequest
    - BillingAddress (AddressDTO)
    - DeliveryAddress (AddressDTO)
    - Title (title enum – allowed values: NOT_SET, Mr, Mrs, Miss, Ms, Dr, Prof, Other) [Unregistered customers]
    - FirstName (string) [Unregistered customers]
    - LastName (string) [Unregistered customers]
    - ContactTelephonePrimary (string) [Unregistered customers]
    - ContactEmailPrimary (string) [Unregistered customers]
    - OrderPayments (array of PaymentDTO)
        - Amount (decimal)

        - PaymentMethodType (enum – allowed values: NOT_SET, Cash, CreditDebitCard, LoyaltyCard, PayLater, MultiplePayment, BillToRoom, Payment_Substitution, Paypal, Vme)
    - SavePaymentCard (nullable boolean)
    - OrderGeneralNotes (string)
    - OrderNickName (string)
    - CalculateTaxFromPOS (boolean)
    - RestoreBasket (nullable boolean)
    - ShopperIPAddress (string)
    - NumberOfGuests (int 32)
    - TableNumber (string)
    - CheckId (string)
    - TipAmount (decimal)
    - PayCheckAttributes (array of KeyValuePairDTO)
        - KeyValueId (int 32)
        - Key (string)
        - Value (string)

Response

- CheckPaymentResponse
    - OrderStatus (enum -)
    - OrderPromiseTime (datetime)
    - OrderConfirmationNumber (string)
    - MoreInfoRequiredRequest (PaymentMoreInfoRequiredDTO)
        - ProviderAttributes (array of KeyValuePairDTO – see above for properties)
        - ProviderControlAlias (string)

PrintCheck

The PrintCheck() method used to get print existing check from POS.

Input Parameters

Table 174 - GetPrintedCheckRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetPrintedCheckRequest	The get printed check request object

Output Parameters

Table 175 - GetPrintedCheckResponse

Type	Description
GetPrintedCheckResponse	The get printed check response object.

When to use

This method should be called to make to get the check details from the POS. This is a text representation of the customer check.

Example

Request

- Session
    - ApplicationId: "A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA"
    - CultureCode: "en-US"
    - SessionId: "154c74ca-fe66-4871-b3d8-fd217f434402"
- GetPrintedCheckRequest
    - CheckId (string)

Response

- GetPrintedCheckResponse
    - PrintedCheckInfo (string)

GetOpenChecks

The GetOpenChecks() method is used to get open checks from POS.

Input Parameters

Table 176 - GetOpenChecksRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetOpenChecksRequest	The get check data request object.

Output Parameters

Table 177 - GetOpenChecksResponse

Type	Description
GetOpenChecksResponse	The get check data response object.

When to Use

This method should be called to get open checks from POS.

Example

Request

- Session
  - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
  - CultureCode: “en-US”
  - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetOpenChecksRequest
  - EmployeeId (int)
  - StoreId(int)

Response

- GetOpenCheckResponse
  - List <OpenCheckInfoDTO>

GetTipOptions

The GetTipOptions() method is used to get tip options from POS.

Input Parameters

Table 178 - GetTipOptionsRequest

Type	Description
SessionDTO	The current customer session object initiated at the start of the session by calling the session service. This is used to uniquely identify the customer.
GetTipOptionsRequest	The service request base object.

Output Parameters

Table 179 - GetTipOptionsResponse

Type	Description
GetTipOptionsResponse	The get tip options response object.

When to Use

This method should be called to get tip options from POS.

Example

Request

- Session
    - ApplicationId: “A4F9AFDB-2E4E-4A15-BC07-C74F0F7107DA”
    - CultureCode: “en-US”
    - SessionId: “154c74ca-fe66-4871-b3d8-fd217f434402”
- GetTipOptionsRequest

Response

- GetTipOptionsResponse
    - TipOptions string[]

4 Data Contract Definitions

Fault Contracts

Service Fault

Should an unhandled exception occur within the services, a fault contract will be generated and messaged to the calling application. The service fault contract is defined as follows:

Table 180 – Service Fault

Type	Name	Description
string	FaultMessage	A textual description of the fault.
ServiceErrorType	ErrorType	An enumerated value which defines the type of fault being raised.

Request Objects

Cancel Order Request

Table 181 – Cancel Order Request

Type	Name	Description	Is Required
Int	OrderId	The order unique identifier	Yes
String	CancellationComments	Comments about the cancelation (explanation)	yes

Get Basket Remove Discount Request

Table 182 – Get Basket Remove Discount Request

Type	Name	Description	Is Required
int	DiscountTypeId	The identifier of the discount type to be removed	Yes

Get Basket Apply Discount

Table 183 – Get Basket Apply Discount

Type	Name	Description	Is Required
int	DiscountTypeId	The discount type to be applied	Yes

decimal	DiscountAmount	The discount amount	Yes

Get User Account Configurations Request

Table 184 – Get User Account Configurations Request

Type	Name	Description	Is Required
Int	BrandId	The brand item identifier. When provided, this value is used instead of the one in session.	No

Store Search By Address Id Request

Table 185 – Store Search By Address Id Request

Type	Name	Description	Is Required
Int	AddressId	The address unique identifier. When provided, this id will be used instead of the one in session.	No
String	CancellationComments	Comments about the cancelation (explanation)	yes

Create Customer Request

Table 186 – Create Customer Request

Type	Name	Description	Is Required
CustomerDTO	Customer	The customer object	Yes
AddressDTO	CustomerAddress DTO	The customer’s address object	Yes

Select Customer and Address Request

Table 187 – Select Customer and Address Request

Type	Name	Description	Is Required
int	CustomerId	Customer unique identifier	Yes
int	CustomerAddressId	The customer’s address unique identifier	Yes

Int	TradeZoneId	The tradezone id of the customer's address (if address has tradezone make this value is included in the request)	No
bool	ReturnCustomerDetails	Flag to indicate whether or not to retrieve the selected customer's details.	No

Start Request Base

All start request objects inherit from a base object called *StartRequestBase*, thus furnishing each start request object with the following properties:

Table 188 – Start Request Base

Type	Name	Description	Is Required
string	Context	A client optional arbitrary string that will be passed back in the associated response. This can be used for asynchronous calling validation or to pass values through the service for use in the client application when the response is handled.	No

Customer Search Request

Table 189 – Start Request Base

Type	Name	Description	Is required * At least one is required
int	LastName	Customer's last name	No*
string	PhoneNumber	Customer's phone number	No*
string	StreetName	Customer's address street name	No*
String	Email	Customer's primary email	No*
string	TownCity	Customer's address town or city name	No*
string	PostCodeOrZip	Customer's address zip or postal code	No*
CustomerStatusType	Status	The customer status	No*

Create Customer Note Request

Table 190 – Create Customer Note Request

Type	Name	Description	Is required
int	TypeId	Customer note type id	Yes
string	TypeText	Customer note description.	Yes
Int	ReasonCodeId	The reason code id. This is mainly used for complaint type customer notes, in order to set a resolution (i.e. grant a discount).	Yes
String	ReasonCodeText	The reason code description.	Yes
Int	ParentId	The parent customer note identifier. When multiple resolutions have been created for a given customer note, this property will usually contain the id of the latest resolution customer note created. In the case where there is only one resolution, this property will contain the id of the original complaint type customer note.	No
Int	RootParentId	The top-level customer note identifier. Normally, this will always be the id of the original complaint type customer note.	No
Int	OrderId	The order id associated to the customer note item (for order complaint type customer notes).	No
string	NoteText	Customer note comments	Yes
bool	IsProcessed	Indicates if the customer note has been resolved. This is usually used for complaint type customer notes to indicate if the complaint has been resolved either with a discount, a callback, or any other kind of resolution.	Yes

string	OrderReference	The reference identifier of the associated order.	No

Get Customer Note Types Request

Table 191 – Get Customer Note Types Request

Type	Name	Description	Is Required
CustomerNoteTypeClass	CustomerNoteTypeClass	The customer note type class	Yes
Int	BrandId	The brand item identifier, if provided, this id is used instead of the one in session.	No

Get Customer Notes Request

Table 192 – Get Customer Notes Request

Type	Name	Description	Is Required
Bool	IsSimplifiedView	Set to true to retrieve the number of Customer Notes indicated by NumberOfNotesToDisplay property, CustomerNote Class filter is not used; Set to false to retrieve only CustomerNotes of the CustomerNote Class. Use OrderId and DaysToShow properties to delimit results.	Yes

Get Customer Notes Request Base

Table 193 – Get Customer Notes Request Base

Type	Name	Description	Is Required
Int	CustomerId	The customer identifier. If provided, this id is used instead of the customer id in session.	No
Int	NumberOfNotesToDisplay	Max number of notes to retrieve.	No
GetCustomerNotesOptions	Option	Defines the kind of notes to retrieve.	Yes
Int	RootParentId	The identifier of the parent customer note (to retrieve child only notes).	No
Int	OrderId	The order id associated with the customer note.	No
Int	DaysToShow	Delimit the number of days to consider when searching for customer notes.	No

Search Open Order Request

Table 194 – Search Open Order Request

Type	Name	Description	Is Required
int	StoreId	Store identifier	Yes
OrderClass	OrderClass	The order class	Yes
bool	ShowOnLine OrderOnly	Flag to indicate if only on line orders will be retrieved	Yes

Resolve Callback Request

Table 195 – Resolve Callback Request

Type	Name	Description	Is Required
CallbackResolution	CallbackResolution	The callback resolution	Yes
int	OrderId	Order item identifier	Yes
string	ReasonCodeId	Resolution reason code id	Yes
string	ReasonCodeDescription	Resolution reason description	Yes
string	Comments	Resolution comments	Yes

Search Order Callback Request

Table 196 – Search Order Callback Request

Type	Name	Description	Is Required
string	LastName	Customer's last name	No
string	OrderReference	Callback order item reference	No
string	First Name	Customer's first name	No
Int	CallbackReasonCodeId	The callback reason code id to search for	No

Search Order Request

Table 197 – Search Order Request

Type	Name	Description	Is Required
string	PhoneNumber	Customer’s phone number	No
string	LastName	Customer’s last name.	No
string	OrderReference	Order item reference	No
string	POSConfirmationNumber	Order item PosConfirmationNumber	No
string	OrganisationName	The organization name	No
string	StreetName	Order item street name (it could be either the customer address street or the store address street, depending on the order type)	No
Datetime	DateFrom	Search from date	Yes
Datetime	DateTo	Search up to date	Yes
OrderStatus ClassType	OrderStatusClass	The order item status class	No
OrderStatus Type	OrderStatus	The order item status	No
Int	SalesChannel	The sale channel item id	No

Get Order Details Request Base

Table 198 – Get Order Details Request Base

Type	Name	Description	Is Required
Int	OrderId	The order item id	Yes

Abandon Call Request

Table 199 – Abandon Call Request

Type	Name	Description	Is Required
Int	ReasonCodeId	The resolution reason code id to be set to the call item.	Yes
string	ReasonCodeDescription	The reason code description and/or comments.	Yes

Get Reason Codes Request

Table 200 – Get Reason Codes Request

Type	Name	Description	Is Required
ReasonCodeType	ReasonCodeType	The ReasonCodeType (enum). Required when *Option* is *GetByReasonCodeType.*	Yes
int	CustomerNoteTypeId	Defines what kind of customer note type will be used to get the reason codes. Required when *Option* is *GetByCustomerNoteTypeId*	Yes
GetReasonCodeOptions	Option	Defines which criteria to use to find the ReasonCodes(enum)	Yes
Int	BrandId	Include the brand id only when no brand has been selected before (brand in session has not been set). The brand id will not be persisted in Session.	No

Select Brand Request

Table 201 – Select Brand Request

Type	Name	Description	Is Required
int	BrandId	The brand item identifier	Yes

Resolve Call Get Next Request

Table 202 – Resolve Call Get Next Request

Type	Name	Description	Is Required
int	CallId	The call item identifier	Yes
int	ReasonCodeId	The resolution reason code id (See GetReasonCodes method for more information)	Yes
string	ReasonCodeDescription	The resolution reason code id description	Yes
bool	ResolveAllCalls	Flag to indicate whether or not to resolve all of the existing open calls for the user in session in one single operation (same reason for all of them will be set)	Yes

Login User Start Session Request

Table 203 – Login User Start Session Request

Type	Name	Description	Is Required
string	Context	A client optional arbitrary string that will be passed back in the associated response. This can be used for asynchronous calling validation or to pass values through the service for use in the client application when the response is handled.	No
string	UserName	The user name	Yes
string	Password	The user name password	Yes
bool	CreatePersistentCookie	Indicates whether or not to create a persistent cookie	False
String	ApplicationId	The id of the application to which the user is signing in	Yes
String	CultureCode	The culture code to be used	yes

Update User Account Request

Table 204 – Update User Account Request

Type	Name	Description	Is Required
UserAccountDTO	UserAccount	An object used to represent the customer’s information.	Yes

Add AddressRequest

Table 205 – Add AddressRequest

Type	Name	Description	Is Required
AddressDTO	Address	An object used to represent the customer’s location.	Yes

Apply Voucher Request

Table 206 – Apply Voucher Request

Type	Name	Description	Is Required
String	VoucherCode	The voucher code.	Yes

Get Basket Add Product Request

Table 207 – Get Basket Add Product Request

Type	Name	Description	Is Required
Int	Quantity	The quantity of the selected product to add to the basket. Note that this is only applicable when adding a menu product Id (See ProductType property below). Default quantity is 1.	Yes
Int	ProductType	BasketProductAddType enumeration value representing the type of product being added to the current basket. This will either be a menu product, a wishlist or a previous order.	No
Bool	IsProductUpsellQualifier	Mark the item to add to basket is an offer's qualifier or not. Default is false.	No
Bool	IsProductUpSellOfferItem	Mark the item to add to basket is an offer's offer item or not. Default is false.	No

Int	ProductUpSellOfferId	The offer's id. If this item to add to the basket is a qualifier or offer item, the offer id should be not null.	No
Int	ProductUpSellOfferGroupId	The offer's group id. If this item to add to the basket is an offer item, the group id should be not null.	No
ContainerState DTO	ContainerStateDTO	Represents related products that are added to the basket together with parent item	No
Int	Inviteeid	The invitee identifier.	No
String	InviteeName	The name of Invitee	No
ContainerState DTO	ContainerStateANPDTO	Container state ANN DTO if it is complex product.	No
ContainerState DTO	ContainerStateMissingDefaultDTO	Container state missing default DTO if it is complex product.	No
Decimal	PriceOverride	Overridden price for product.	No
Bool	CreateNewBasket	Flag if product need to be added to new basket.	No
Bool	AllowContainerToBasket	Flag whether to allow container to basket whilst rendering.	No
DateTime	OrderTime	The time of order	No

Get Basket Add Multiple Products Request

Table 208 – Get Basket Add Multiple Products Request

Type	Name	Description	Is Required
List<BasketProductDTO>	BasketProductCollection	Represent list of basketproduct dto that are added to the basket	Yes
Bool	CreateNewBasket	Flag if product need to be added to new basket.	No
Bool	AllowContainerToBasket	Flag whether to allow container to basket whilst rendering	No
DateTime	OrderTime	The time of order	No

Decline Current Product Upsell Request

Table 209 – Decline Current Product Upsell Request

Type	Name	Description	Is Required
Int	Inviteeid	The invitee identifier	No
String	InviteeName	The name of Invitee	No

Get Customer Favorites Request

Table 210 – Get Customer Favorites Request

Type	Name	Description	Is Required
bool	IncludeFavoriteItems	value indicating whether [include order items]	No

Get Customer Favorites Response

Table 211 – Get Customer Favorites Response

Type	Name	Description	Is Required
List<FavoriteDTO>	Favorites	List of favorite that was saved by user	

Remove Favorite Request

Table 212 – Remove Favorite Request

Type	Name	Description	Is Required
int	FavoriteId	The basket Id which is also a wishlist's Id	No
bool	IsDeleteFavorite	Specify delete favorite or individual items	Yes
List<int>	FavoriteItemIds	Specify favorite items that will be removed	No

Save Favorite Request

Table 213 – Save Favorite Request

Type	Name	Description	Is Required
String	WishlistName	The name of the wishlist	No
Bool	IsSaveBasket	Specify save whole basket or individual items	Yes
List<int>	BasketItemIds	List of Basket Item will be saved as new favorite or add to existed favorite	No
Int	FavoriteId	Id of favorite to save into	No

Save Favorite Response

Table 214 – Save Favorite Response

Type	Name	Description	Is Required

Check Product Availability Request

Table 215 – Check Product Availability Request

Type	Name	Description	Is Required
List<int>	ProductIds	The product Ids that need to verify.	Yes
Bool	OrderTime	Specify Order time that will use to verify product ids	Yes
List<int>	StoreId	Specify Store Id that will use to verify product ids	Yes

Check Product Availability Response

Table 216 – Check Product Availability Response

Type	Name	Description	Is Required
List<int>	NotAvailableProductIds	The product Ids that are not available.	Yes

Get Basket Remove Item Request

Table 217 – Get Basket Remove Item Request

Type	Name	Description	Is Required
int	BasketItemId	The BasketItemId corresponding to the basket item selected by the customer to be removed from the basket. This value is obtained from the BasketItemId property of the BasketItem object.	Yes

Get Basket Clear Partial Request

Table 218 – Get Basket Clear Partial Request

Type	Name	Description	Is Required
int	InviteeId	The Invitee identifier.	No

Get Basket Clear Items Request

Note that this is currently an empty object.

Table 219 – Get Basket Clear Items Request

Type	Name	Description	Is Required

Get Basket Update Quantity Request

Table 220 – Get Basket Update Quantity Request

Type	Name	Description	Is Required
List<BasketItemSummaryDTO>	BasketItemsToUpdate	List of basket item summary DTO with updated information for basket items in current basket	No

Get Basket Request

Note that this is currently an empty object. The basket for the current user is managed within the services and can be returned via information provided about the current user session.

Table 221 – Get Basket Request

Type	Name	Description	Is Required
bool	IsTaxProcessingRequired	Flag indicating whether tax processing required on the returned basket. If tax processing is run, the basket will contain a basketTaxes collection if applicable. The tax charges are usually only displayed at the point of checkout to minimize load on the server. Bear in mind that tax calculations may require calling the store POS so this can add a large overhead, so please only use this when required.	No
bool	IsSurchargeProcessingRequired	Flag indicating whether surcharge processing is required on the returned basket. If surcharge processing is run, the basket will contain a basketSurcharges collection if applicable. The surcharges are usually only displayed at the point of checkout to minimize load on the server. Bear in mind that surcharge calculations may require calling the store POS so this can add a large overhead, so please only use this when required.	No

PaymentDTO	PaymentDetails	The details of the current payment type for the order. This may be required to process calculations for payment based surcharges.	No

Validate Basket Against Changes Request

Table 222 – Validate Basket Against Changes Request

Type	Name	Description	Is Required
Int	StoreId	The id of store	Yes
DateTime	OrderTime	New order time if order time is changed.	No
FulfillmentTimeType	FulfillmentTimeType	New fulfillment time type if fulfillment time type is changed.	No

Confirm Validate Basket Against Changes Request

Table 223 – Confirm Validate Basket Against Changes Request

Type	Name	Description	Is Required
bool	RollbackChanges	RollbackChanges = true: revert to previous state (before ValidateBasketAgainstChanges get called) To confirm the changes, RollbackChanges = false	False

Get Bucket Configurator Request

Table 224 – Get Bucket Configurator Request

Type	Name	Description	Is Required
Int	ProductId	The identifier of the root product	Yes
Int	BasketItemId	The basket identifier of the product selection	No

Update Bucket Item Request

Table 225 – Update Bucket Item Request

Type	Name	Description	Is Required
BucketConfiguration StateDto	CurrentConfigurationState	The current state of the product being configured.	Yes
Int	RootProductId	The identifier of the overall root product being configured.	Yes
Int	ProductId	The identifier of the step item selected	Yes
Int	StepProductId	The identifier of the containing step product	Yes
Int	QuantitySelected	The quantity selected of the step item selected	Yes
Bool	ExclusiveSelection	An indicator identifying whether the selection is part of an exclusive group	Yes

Complete Bucket Configuration Request

Table 226 – Complete Bucket Configuration Request

Type	Name	Description	Is Required
BucketConfiguration StateDto	CurrentConfigurationState	The current state of the product being configured.	Yes
Int	BasketItemId	The identifier of the basket containing the completed configuration	No

Get Content Request

Table 227 – Get Content Request

Type	Name	Description	Is Required
string	ContentName	Name of the Content to retrieve	Yes
string	HierarchyLocation	At what level of the hierarchy to retrieve content. If not found at the current level then attempt to find at the ancestor level	Yes
Byte[]	LastTimeStamp	Optional parameter returned from last call to retrieve the same content.	No

Get Customer Account Request

Table 228 – Get Customer Account Request

Type	Name	Description	Is Required
bool	GetCustomerAddressDetails	A boolean flag which determines whether to include stored customer address information in the returned message.	Yes
bool	GetCustomerPaymentCardDetails	A Boolean flag which determines whether to include stored customer payment card details in the returned message.	Yes

GetMenuAndStartOrderRequest

Table 229 – GetMenuAndStartOrderRequest

Type	Name	Description	Is Required
int	StoreId	The StoreId corresponding to the store selected by the customer. This value is obtained from the StoreId property of the StoreDTO object.	Yes
OrderClass	OrderClass	An OrderClass enumeration value representing the type of order (collection, delivery). This value should be contained within the SupportedOrderTypes collection of the Store DTO object.	Yes
FulfillmentTime Type	FulfillmentTime Type	A value from the Fulfillment Time Type enumeration to represent the order fulfillment time type (ASAP, advanced, future).	Yes
DateTime	OrderTime	The date/time the order is required to the nearest minute. This value must be an available order time at the selected store. Available order times are provided in the AvailableOrderTimes collection of the Store DTO object.	Yes
int	CurrentMenuId	The MenuId corresponding to the current menu in use by the client application. This can be set to prevent the menu being returned if it has not altered as a result of the request.	No
bool	RemoveEmptyC ategory	Flag to remove all empty categories from response.	No
Int	CustomerId	When provided, the method will use this identifier instead of the one in Session (useful in a call center context)	No

Get Store Order Options Request

Table 230 – Get Store Order Options Request

Type	Name	Description	Is Required
int	StoreId	The StoreId corresponding to the currently selected store.	Yes
DateTime	OrderTime	The date the order is required. This value is used to calculate the order times available on this date. At present only current day orders are supported, but this will be used for ordering on a future date in later versions of the service.	Yes
bool	NormalView	Flag for normal view	No
bool	UseStoreLocalTime	Flag to use store local time for store order options.	No

Login Request

Table 231 – Login Request

Type	Name	Description	Is Required
string	UserName	The UserName of the user logging in.	Yes
string	Password	The Password of the user logging in.	Yes
String	CreatePersistentCookie	Flag to determine whether a persistent cookie should be created.	Yes

Remove Customer Payment Card Request

Table 232 – Remove Customer Payment Card Request

Type	Name	Description	Is Required
Int	CustomerPaymentCardId	Customer payment card identifier	Yes

Password Recovery Request

Table 233 – Password Recovery Request

Type	Name	Description	Is Required
string	UserName	The UserName of the user requesting their password.	Yes

Place Order Request

Table 234 – Place Order Request

Type	Name	Description	Is Required
AddressDTO	BillingAddress	An object used to represent the billing address for current order.	No
AddressDTO	DeliveryAddress	An object used to represent delivery address for current order.	No
PaymentDTO	OrderPayments	A list of objects representing one or more payments provided for the order to be processed.	Yes
string	OrderGeneralNotes	An optional text value to place as notes against the order.	No
string	OrderNickName	An optional text value to provide a name to the order being placed, allowing recovery for repeat orders in the future.	No
String	Title	Title of un-registered customer	No
String	FirstName	First name of un-registered customer	No
String	LastName	Last name of un-registered customer	No

String	ContactTelephonePrimary	Primary telephone number for un-registered customer	No
String	ContactEmailPrimary	Primary email address for un-registered customer	No
Bool	MarketingOptIn	Flag indicating whether user has opted-in for marketing messages.	No
Bool	SavePaymentCard	Flag indicating whether payment card details to be saved in mycental database.	No
Int	ChannelPaymenMethodId	Channel payment method identifier.	No
Bool	CalculateTaxFromPOS	Flag indicating whether tax calculations to be done from POS.	No
Bool	RestoreBasket	Flag indicating whether to restore basket.	No
String	ShopperIPAddress	Shopper's IP address	No
Int	NumberOfGuests	Number of guests	No
String	ReasonCode	Callback reason identifier for future ordering	No
Sting	CheckId	The table number	No
List<KeyValuePair DTO>	PlaceOrderAttributes	List of key value pair attributes for current order.	No
Long	FaceBookUserId	Facebook user identifier	No
Bool	ConfirmOrder	Flag indicating whether order if confirmed.	No
String	OrderName	Order named by customer so they can recognize each order independently in future	No

Register Request

Table 235 – Register Request

Type	Name	Description	Is Required
CustomerDTO	Customer	An object used to represent the customer.	Yes
AddressDTO	Address	An object used to represent the customer's location, used as the default address of the registering customer.	Yes
string	Password	The Password for the registering user.	Yes

Remove Address Request

Table 236 – Remove Address Request

Type	Name	Description	Is Required
int	CustomerAddressId	The AddressId corresponding to the customer address selected by the customer. This value is obtained from the AddressId property of the AddressDTO object.	Yes
Int	CustomerId	When provided, the method will use this identifier instead of the one in Session (useful in a call center context).	no

Start Session Request

Table 237 – Start Session Request

Type	Name	Description	Is Required
string	ApplicationId	A unique key used for calling application.	Yes
string	CultureCode	A valid language culture name configured in OHEICS for the client brand. (see http://msdn.microsoft.com/en-us/library/ms866170.aspx for a complete list)	Yes

Store Search Request

Table 238 – Store Search Request

Type	Name	Description	Is Required
AddressDTO	Address	An object used to represent the customer's location.	Yes
DateTime	OrderTime	The date the order is required. This value is used to calculate the order times available on this date.	Yes
String	AccountCode	Account code for store search	No
int	ClientOrderType	Client order type (i.e. Collection or Delivery) identifier.	No
bool	SortByBusinessDay	Flag to sort search result by business day.	No
bool	NormalView	Flag to get normal view of stores.	No
int	StoreAttributeId	Filter and return stores which support given store attribute. i.e. Delivery, Collection, Dine-in, etc.	No
bool	UseStoreSearchProvider	Flag to indicate if the search process will be performed using the StoreSearchProvider (Brand StoreSearchProvider) instead of the GeoCodeSearch	No
bool	IncludeOrderTypeOpening Times	Indicates whether Order Type Times will be included or not in the store opening times	No

Store List Request

Table 239 – Store List Request

Type	Name	Description	Is Required
Int	PageNumber	Current page number for store list	Yes
Int	PageSize	Number of stores per page	Yes
DateTime	OrderTime	The order time	Yes
bool	NormalView	Flag for normal view	No
Int	OrderClass	Specify the desired OrderClass (pickup, delivery etc) in session state	Yes

Store Names Request

Table 240 – Store Names Request

Type	Name	Description	Is Required
Int	RegionId	An optional parameter to narrow the result set to stores within a region.	No

Store Detail Request

Table 241 – Store Detail Request

Type	Name	Description	Is Required
Int	StoreId	Store identifier	Yes
DateTime	WeekStartDateTime	Business start date time of week	No
bool	sortByBusinessStartDay	Sort results by business start day	No

Store Attribute Request

Table 242 – Store Attribute Request

Type	Name	Description	Is Required
Int	BrandId	Brand identifier	No
Int	StoreAttributeTypeId	Store attribute type identifier	No

Update Address Request

Table 243 – Update Address Request

Type	Name	Description	Is Required
AddressDTO	Address	An object used to represent the customer’s location.	Yes

Update Customer Request

Table 244 – Update Customer Request

Type	Name	Description	Is Required
CustomerDTO	Customer	An object used to represent the customer.	Yes
bool	UpdateAsSecondPerson	If true, it means that a second person (i.e. an operator in a Callcentre context) is updating the Customer account, otherwise, false which means, the customer itself is updating its own account (use this in a non-call center context, this is the default behavior).	No

Address Search Request

Table 245 – Address Search Request

Type	Name	Description	Is Required
AddressDTO	Address	An object used to contain the input parameters for the address search.	Yes
int	StartPage	Indicates the page number of the result set to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No
int	PageSize	Indicates the size of the result set page to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No

Get Customer Orders Request

Table 246 – Get Customer Orders Request

Type	Name	Description	Is Required
Bool	IncludeOrderItems	Flag indicating whether to include order items	No
CustomOrderFilterOptions	OrderFilterOption	Filter option to get All, Recent or wishlist orders.	No
DateTime	AfterDate	The all of orders in the result list will be greater or equal this date. Just provide local time.	No
DateTime	BeforeDate	The all of orders in the result list will be less or equal this date. Just provide local time.	No
int	CurrentPageIndex	This is the page index at current. It's start from 0 at first page. If CurrentPageIndex is null or less 0, then its value will be set to0.	No
int	PageLength	It specifies the page size. If parameter is null then it will be set as the value of AppSettings by key word "RecentOrdersDisplay" in database; If it did not exist in AppSettings, then the value will be set to 5.	No

Save Basket As Wishlist Request

Table 247 – Save Basket As Wishlist Request

Type	Name	Description	Is Required

GetConfigurator Request

Table 248 – GetConfigurator Request

Type	Name	Description	Is Required
ProductConfigurationActions	SelectedAction	An enumeration of various actions e.g. BaseSelected, SizeSelected etc.	Yes
int	Quantity	Indicates the quantity required for the initial selected product. Defaults to 1 where absent.	No
int	Id	Id field of selected item. This will be the Id of the product initially selected and the Id of subsequently selected ingredients	Yes
int	ParentProductToppingGroupId	This field is required when selecting a topping.	No
int	ClassId	This field is required when selecting a topping.	No
Int	BasketItemId	The basket item identifier, if product is being added to then it would be null. Basket item id will have value is basket is being edit.	No
Bool	ReturnAllSections	Flag to decide whether to include list of configuration states and options.	No
Int	DealProductId	Product Id of the Deal.	No
Int	DealStepId	Identifier of deal step if product is a deal.	No
List<ConfigurationActionDTO>	ConfigurationActions	The list of configuration actions.	No

Bool	FilterSpecialitiesForSelectedBaseAndSize	Flag indicating whether to validate and filter specialties for selected base and size or not. If this set to true core will check all specialties for complex product and return only specialties where selected base and size is supported. Currently this is being used Mobile website only as there user can select different specialty for each half of the pizza.	No
Bool	IsProductUpsellQualifier	Mark the item to add to basket is an offer's qualifier or not. Default is false.	No
Bool	IsProductUpSellOfferItem	Mark the item to add to basket is an offer's offer item or not. Default is false.	No
Int	ProductUpSellOfferId	The offer's id. If this item to add to the basket is a qualifier or offer item, the offer id should be not null.	No
Int	ProductUpSellOfferGroupId	The offer's group id. If this item to add to the basket is an offer item, the group id should be not null.	No

Configure Product Request

Table 249 – Configure Product Request

Type	Name	Description	Is Required
Int	ProductId	Product Identifier to load product	Yes
int	Quantity	Quantity of product	No
ProductConfigurationStateDTO[]	ConfigurationSelection	Selections of configuration for each section.	Yes

Deal Completed Request – Inherits GetBasketRequest

Table 250 – Deal Completed Request – Inherits GetBasketRequest

Type	Name	Description	Is Required
Bool	IsCancelled	Flag indicating whether this instance of deal has been cancelled.	No
int	Quantity	Indicates the quantity required for the initial selected product. Defaults to 1 where absent.	No
List<ProductConfiguration KeyValuePairDTO>	Selected Products	List of selected products for this deal. This list will contain one entry for each step in deal. All steps must have an entry, complex, container or simple products.	Yes
Int	BasetItemId	The basket item identifier for deal if basket is being edited.	
Bool	IsProductUpsellQu alifier	Mark the item to add to basket is an offer's qualifier or not. Default is false.	No
Bool	IsProductUpSellOff erItem	Mark the item to add to basket is an offer's offer item or not. Default is false.	No
Int	ProductUpSellOffe rId	The offer's id. If this item to add to the basket is a qualifier or offer item, the offer id should be not null.	No
Int	ProductUpSellOffe rGroupId	The offer's group id. If this item to add to the basket is an offer item, the group id should be not null.	No

Get Deal Configurator Request

Table 251 – Get Deal Configurator Request

Type	Name	Description	Is Required
int	ProductId	Product identifier of root product in a deal.	Yes
int	Quantity	Indicates the quantity required for the initial selected product. Defaults to 1 where absent.	No
Int	BasketItemId	The basket item identifier, if product is being added to then it would be null. Basket item id will have value is basket is being edit.	No

Get Payment Options Request

Table 252 – Get Payment Options Request

Type	Name	Description	Is Required
Int	GroupOrderType	Type of group order	No

Get Checkout Payment Options Request

Table 253 – Get Checkout Payment Options Request

Type	Name	Description	Is Required
Int	GroupOrderType	Type of group order	No

Place Order Complete Payment Request

Table 254 – Place Order Complete Payment Request

Type	Name	Description	Is Required
Payment MoreInfo DTO	MoreInforResponse	MoreInfo Response object from placeorder method.	No

Place Order Request

Table 255 – Place Order Request

Type	Name	Description	Is Required
AddressDTO	BillingAddress	An address object available for the provision of an alternative billing address	No
AddressDTO	DeliveryAddress	The full delivery address for the order	No
Title	Title	The customer's title from the enumerator'Title' type	No
string	FirstName	The customer's first name	No
string	LastName	The customer's last name	No
string	ContactTelephonePrimary	The customer's primary means of telephone contact.	No
string	ContactEmailPrimary	The customer's primary means of email contact.	No
PaymentDTO	OrderPayments	An array of payments	Yes
string	OrderGeneralNotes	Not currently in use.	No
string	OrderNickName	Not currently in use.	No
bool	IsCallbackRequired	Flag to indicate whether or not the order requires callback (i.e. supervisor approval).	No
Int	ReasonCodeId	The reason code id for the callback (see GetReasonCodes method). *This property becomes required when IsCallbackRequired flag is true.	No*
String	CallbackComments	Comments about the callback (explanation)	No*
List<int>	OrderOptions	Order options to be set to the order (See GetOrderOptions method)	No

Activate Loyalty Account Request

Table 256 – Activate Loyalty Account Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Check Loyalty Account Balance Request

Table 257 – Check Loyalty Account Balance Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Create Loyalty Account Request

Table 258 – Create Loyalty Account Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Credit Loyalty Account Request

Table 259 – Credit Loyalty Account Request

Type	Name	Description	Is Required
decimal	Amount	The currency or points amount to credit	Yes
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Get Loyalty Account Transaction History Request

Table 260 – Get Loyalty Account Transaction History Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Get Loyalty Coupon List Request

Table 261 – Get Loyalty Coupon List Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

List Loyalty Accounts Request

Table 262 – List Loyalty Accounts Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes
Bool	AccountBalance	Balance of account	No

Validate Loyalty Account Request

Table 263 – Validate Loyalty Account Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Link Existing Account Request

Table 264 – Link Existing Account Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Lost Loyalty Card Request

Table 265 – Lost Loyalty Card Request

Type	Name	Description	Is Required
KeyValuePairDTO[]		Attributes	An array of key value pair objects

Update Loyalty Account Request

Table 266 – Update Loyalty Account Request

Type	Name	Description	Is Required
CustomerDTO	Customer	Containing information about customer	No
AddressDTO	CustomerAddress	Containing information about customer address	No
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Get Unique Items Request

Table 267 – Get Unique Items Request

Type	Name	Description	Is Required
KeyValuePairDTO[]	Attributes	An array of key value pair objects	Yes

Apply Voucher Request

Table 268 – Apply Voucher Request

Type	Name	Description	Is Required
string	VoucherCode	The code of the voucher to be applied	Yes

Get Basket Apply Voucher Request

Table 269 – Get Basket Apply Voucher Request

Type	Name	Description	Is Required
string	VoucherCode	The code of the voucher to be applied	Yes
Other properties		This inherits from GetBasketRequest and therefore contains the same properties	

Remove Voucher Request

Table 270 – Remove Voucher Request

Type	Name	Description	Is Required
		Contains nothing – reserved for future expansion	

Get Basket Remove Voucher Request

Table 271 – Get Basket Remove Voucher Request

Type	Name	Description	Is Required
		This inherits from GetBasketRequest and therefore contains the same properties. It has no additional properties	

Get Basket Add Voucher Request

Table 272 – Get Basket Add Voucher Request

Type	Name	Description	Is Required
		Contains nothing – reserved for future expansion	

Get Order Confirmation Request

Table 273 – Get Order Confirmation Request

Type	Name	Description	Is Required
		Contains nothing – reserved for future expansion	

Get BasketAddProductOffer Request

Table 274 – Get BasketAddProductOffer Request

Type	Name	Description	Is Required
integer	GroupId	The Id of the group that the product to be added is in	Yes
Integer	ProductId	The Productd of the product that is to be added	Yes
Int	InviteeId	The Invitee identifier	No
String	InviteeName	The name of Invitee	No

Add Invitee Request

Table 275 – Add Invitee Request

Type	Name	Description	Is Required
Bool	CustomerValidationRequired	The Boolean flag which defines whether the attached invitee object needs to be validated or not.	Yes
InviteeDTO	GrouporderInvitee	The object used to represent the Invitee	Yes

Get Grouporder By Customer Request

Table 276 – Get Grouporder By Customer Request

Type	Name	Description	Is Required
integer	CustomerId	The logged on CustomerId	Yes

Update Status Request

Table 277 – Update Status Request

Type	Name	Description	Is Required
Integer	CustomerId	The logged on CustomerId	No
String	Status	The new Status of the Invitee (Possible Values: EmailSent, Created, Deleted, EmailFailed, ReminderSent, ReminderFailed , CheckOut)	Yes
Guid	GUID	Unique Identifier that is assigned to each inviteeinstance when the system sends the invitation email.	Yes
Bool	IsEC	The Boolean flag indicates whether the request is from the event coordinator	Yes

Add Grouporder Request

Table 278 – Add Grouporder Request

Type	Name	Description	Is Required
GroupOrderDTO	GroupOrder	The object used the represent the GroupOrder	Yes

Getgrouporder By Id Request

One of the following value should be passed to retrieve the GroupOrder Information.

Table 279 – Getgrouporder By Id Request

Type	Name	Description	Is Required
Integer	GroupOrderId	The GroupOrderId of the GroupOrder that needs to be retrieved	No
String	GroupOrderName	Name of the GroupOrder that needs to be retrieved	No

Add Inviteegroup Request

Table 280 – Add Inviteegroup Request

Type	Name	Description	Is Required
InviteeGroupDTO	InviteeGroup	The object used the represent the InviteeGroup	Yes

Add Invitee to Inviteegroup Request

Table 281 – Add Invitee to Inviteegroup Request

Type	Name	Description	Is Required
List<InviteeDTO>	InviteeList	The List of Invitee objects that needs to be added to the invitee group	Yes
Integer	InviteeGroupId	The InviteeGroupId to which the invitees to be added	Yes
Bool	CustomeValidationRequired	The Boolean flag which defines whether the attached invitee object needs to be validated or not.	Yes

Getgrouporder By Guid Request

Table 282 – Getgrouporder By Guid Request

Type	Name	Description	Is Required
Guid	Guid	The GUID that will be passed in the guest invitation email	Yes

Reset Session Order Details Request

Table 283 – Reset Session Order Details Request

Type	Name	Description	Is Required

Save Product Review Request

Table 284 – Save Product Review Request

Type	Name	Description	Is Required
Int	UserId	The user identifier	Yes
Int	ProductId	The product Identifier.	Yes
String	Comment	Comments about rating	No
String	HierarchicalStructId	Hierarchical structure identifier for the product.	No
Decimal	Rating	User's rating value for the product	Yes
Bool	Flag	The flag for rating	No

Find Alternative Products Request

Table 285 – Find Alternative Products Request

Type	Name	Description	Is Required
Int	ProductId	The product Identifier.	Yes
Int	Quantity	The quantity of the product	Yes
Bool	PerformCloseMatch	Flag indicating whether to perform a close or exact match	No

Get Product By Product Id Request

Table 286 – Get Product By Product Id Request

Type	Name	Description	Is Required
Int	ProductId	The product Identifier.	Yes

Get Product By Category Request

Table 287 – Get Product By Category Request

Type	Name	Description	Is Required
Int	categoryId	The category Identifier.	Yes
Int	SubCategoryId	The sub category identifier	No

User Account Status Request

Table 288 – User Account Status Request

Type	Name	Description	Is Required
String	UserName	Name of the user, this is usually a customer's email address.	Yes

Forgot Password Request

Table 289 – Forgot Password Request

Type	Name	Description	Is Required
String	ConfirmBaseUrl	Url for reset password page, this will be sent through email for user to reset password.	Yes
Decimal	LinkExpireAfterThisTime	Expiry time for reset password link, once this time passed key embedded in base url to reset password won't work.	Yes
String	UserName	The user name	Yes

Reset Password Request

Table 290 – Reset Password Request

Type	Name	Description	Is Required
String	Url	Url for reset password page, this will be sent through email for user to reset password.	Yes
String	Password	New password	Yes
String	RetypedPassword	Retyped new password	Yes
Guid	ResetRequestId	Unique identifier for reset password request generated by OHEICS	Yes

Address Validation Request

Table 291 – Address Validation Request

Type	Name	Description	Is Required
AddressDTO	Address	Address DTO to validate	Yes

Get Address Type Ahead TownCity by Region Request

Table 292 – Get Address Type Ahead TownCity by Region Request

Type	Name	Description	Is Required
AddressDTO	Address	An object used to contain the input parameters for the address search.	Yes
int	StartPage	Indicates the page number of the result set to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No
int	PageSize	Indicates the size of the result set page to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No

Get Address Type Ahead Street By TownCity Request

Table 293 – Get Address Type Ahead Street By TownCity Request

Type	Name	Description	Is Required
AddressDTO	Address	An object used to contain the input parameters for the address search.	Yes
int	StartPage	Indicates the page number of the result set to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No
int	PageSize	Indicates the size of the result set page to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No

Get Address Type Ahead District By TownCity Request

Table 294 – Get Address Type Ahead District By TownCity Request

Type	Name	Description	Is Required
AddressDTO	Address	An object used to contain the input parameters for the address search.	Yes
int	StartPage	Indicates the page number of the result set to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No
int	PageSize	Indicates the size of the result set page to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No

Get Address Type Ahead Region Request

Table 295 – Get Address Type Ahead Region Request

Type	Name	Description	Is Required
int	StartPage	Indicates the page number of the result set to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No
int	PageSize	Indicates the size of the result set page to return when paged results are required. If this value is missing or zero then the full result set of the query will be returned.	No

Get Address Type Ahead Addresses Request

Table 296 – Get Address Type Ahead Addresses Request

Type	Name	Description	Is Required
AddressSearchtype[]	SearchReturnAddressFields	List of address fields to be returned by this method	Yes
AddressSearchtype[]	SearchByAddressFields	List of address fields to be used as search reference fields.	Yes
String[]	SearchAddressValues	List of address fields values specified in SearchByAddressFields property	Yes

Authenticate Employee Request

Table 297 – Authenticate Employee Request

Type	Name	Description	Is Required
Int	EmployeeId	Employee Identifier to be authenticated.	Yes
Int	StoreId	Store Identifier for which employee id is authenticated.	Yes

SignOut Employee Request

Table 298 – SignOut Employee Request

Type	Name	Description	Is Required
Int	EmployeeId	Employee Identifier.	Yes
Int	StoreId	Store Identifier.	Yes

Get Client Configuration Request

Table 299 – Get Client Configuration Request

Type	Name	Description	Is Required
String	Applicationid	Client application identifier	No
String	ApplicationUrl	Client application url	No

Get Client Local Configuration Request

Table 300 – Get Client Local Configuration Request

Type	Name	Description	Is Required

Get Deal Configuration By Key Request

Table 301 – Get Deal Configuration By Key Request

Type	Name	Description	Is Required
String	Key	Configuration key for the deal	Yes

Get Configurations Request

Table 302 – Get Configurations Request

Type	Name	Description	Is Required
ConfigurationTypeEnum	ConfigurationType	Type of configuration	No
Int	StoreId	Store identifier, if null then used from session state	No
Int	Brandid	Brand identifier, if null then used from session state	No

Get Localization Configurations Request

Table 303 – Get Localization Configurations Request

Type	Name	Description	Is Required
ConfigurationTypeEnum	ConfigurationType	Type of configuration	No
OrderClass	OrderClass	The order class	No

Map Localization Configuration To Address Request

Table 304 – Map Localization Configuration To Address Request

Type	Name	Description	Is Required
LocalisationConfigurationDTO	Configuration	Address localization configuration	No

Map Address To Localization Configuration Request

Table 305 – Map Address To Localization Configuration Request

Type	Name	Description	Is Required
LocalisationAddressCategory	AddressCategory	The Address category	No
OrderClass	OrderClass	The order class	No
AddressDTO	Address	Address to be mapped	No

Add Money Request

Table 306 – Add Money Request

Type	Name	Description	Is Required
AddressDTO	BillingAddress	The billing address for the payment card	No
PaymentCardDTO	OrderPayment	The order payment details	No
Bool	SavePaymentCard	Flag indicating whether you want to store payment card details	No
Decimal	Amount	Amount to add	No
Bool	NormalView	Normal view	No

Register Request

Table 307 – Register Request

Type	Name	Description	Is Required
CustomerDTO	Customer	Details of the customer	Yes
AddressDTO	CustomerAddress	The customer address	No
String	Password	The password	Yes

Add Customer Paymentcard Request

Table 308 – Add Customer Paymentcard Request

Type	Name	Description	Is Required
PaymentCardDTO	CustomerPaymentCard	The customer payment card details	Yes

Update Customer Paymentcard Request

Table 309 – Update Customer Paymentcard Request

Type	Name	Description	Is Required
PaymentDTO	CustomerPayment	The customer payment card details	Yes

Hide Past Order Request

Table 310 – Hide Past Order Request

Type	Name	Description	Is Required
Int	OrderId	ID of order that needs to be set as hidden	Yes

Response Objects

Service Response Base

All service response objects inherit from a base object called *ServiceResponseBase,* thus furnishing each service response object with the following properties:

Table 311 – Service Response Base

Type	Name	Description
bool	IsSuccess	A Boolean flag which defines the status of the request made by the calling application.
ServiceErrorType	ErrorType	A Service Error Type enumeration representing the type of error which has occurred in the request (where applicable).
string	ErrorMessage	A string representation of the error.
string	Context	A client optional arbitrary string that will be passed through in the associated request. This can be used for asynchronous calling validation or to pass values through the service for use in the client application when the response is handled.

ValidationErrorDTO []	ValidationErrors	A list of Store AttributeDTO

The store attribute dto contains information about supported store attributes.

Table 481 – Store AttributeDTO

Type	Name	Description
int	StoreAttributeId	The unique identifier for store attribute.
StoreAttributeType	StoreAttributeTypeId	Store attribute type id.
string	Display Text	Text to display on screen, This is translatable text.
string	Description	Detailed description of sore attribute.

Store Lite DTO

The store lite dto contains very basic store information, the store id and store name.

Table 482 – Store Lite DTO

Type	Name	Description
int	StoreId	The unique identifier for the store
string	StoreName	The store name.

Store Order Option DTO

The Store Order Option DTO contains information about the available start order options for the restaurant or store. A list of Store Order Option

DTO objects is typically provided, one for each order class (collection, delivery etc). The DTO includes information on time slots available for ordering against.

Table 483 – Store Order Option DTO

Type	Name	Description
int	StoreId	The unique identifier for the store
List< DateTime>	AvailableOrderSlots	A list of time slots available for ordering against.
Order Class	StoreOrderClass	The class of order that the available order slots are applicable to. I.e. collection or delivery.
List< DateTime>	AvailableOrderDateSlots	A list of date slots available for ordering against.

Validation Error DTO objects which define validation issues with the associated request.

Cancel Order Response

Table 312 – Cancel Order Response

Type	Name	Description
CancelOrderResultType	CancelOrderResult	The operation end result

Get Active User Configurations Response

Table 313 – Get Active User Configurations Response

Type	Name	Description
List<BrandDTO>	Brands	List of valid brands
List<ChannelDTO>	Channels	List of valid brands

Get User Account Configurations Response

Table 314 – Get User Account Configurations Response

Type	Name	Description
List<UserTypeDTO>	UserTypes	User types
List<LoginPeriodDTO>	LoginPeriods	Login periods
List<RegionDTO>	Regions	Regions
List<TimeZoneInfoDTO>	TimeZones	TimeZones

Select Customer and Address Response

Table 315 – Select Customer and Address Response

Type	Name	Description
CustomerDetailDTO	CustomerDetails	An object with the customer details

Customer Search Response

Table 316 – Customer Search Response

Type	Name	Description
List<CustomerResultDTO>	CustomerResultList	A list containing the customer objects.

Create Customer Note Response

Table 317 – Create Customer Note Response

Type	Name	Description
int	CustomerNoteId	The identifier of the customer note created

Get Customer Note Type Response

Table 318 – Get Customer Note Type Response

Type	Name	Description
List<CustomerNoteTypeDTO>	CustomerNoteTypes	A list containing the customer note type objects.

Get Customer Complaints Response

Table 319 – Get Customer Complaints Response

Type	Name	Description
List<CustomerNoteDTO>	CustomerComplaints	A list containing the customer complaint note objects.

Get Customer Note Response

Table 320 – Get Customer Note Response

Type	Name	Description
List<CustomerNoteDTO>	CustomerNotes	A list containing the customer note objects.

Get Dashboard Report Response

Table 321 – Get Dashboard Report Response

Type	Name	Description
Int	OrdersTakenLastHour	Number of orders taken within the last hour
Int	OrdersTakenToday	Number of orders taken today
Int	AverageOrdersPerOperator	Average orders per operator
Decimal	AverageCallsPerOperator	Average calls per operator
Int	FailedOrdersInLastHour	Number of failed orders within the last hour
Int	FailedOrdersToday	Number of failed orders today
Int	NumSuccessOrdersToday	Number of successful orders today

Int	OrdersWaitingCallback	Number of orders waiting for callback resolution
Int	RestaurantsOnLine	Number of restaurants on line (available for ordering)
Decimal	AverageOrderValue	Average order total amount
Decimal	TotalOrderValue	Total order value

Search Open Order Response

Table 322 – Search Open Order Response

Type	Name	Description
List<POSOrderSummaryDTO>	Orders	A list containing the order objects.

Validate Callback Mandatory Response

Table 323 – Validate Callback Mandatory Response

Type	Name	Description
bool	IsCallbackMandatory	A flag to indicate whether or not a callback is mandatory for the current order in session
Int	CallbackReasonCodeId	The mandatory reason code id to be used in the callback

Search Order Callback Response

Table 324 – Search Order Callback Response

Type	Name	Description
List<CallOrderCompactDTO>	Orders	A list containing the order objects.

Search Order Response

Table 325 – Search Order Response

Type	Name	Description
List<CallOrderCompactDTO>	Orders	A list containing the order objects.

Get Order Details Response

Table 326 – Get Order Details Response

Type	Name	Description
CallOrderDTO	Order	An object containing the order details

Get Reason Codes Response

Table 327 – Get Reason Codes Response

Type	Name	Description
List<ReasonCodeDTO>	ReasonCodes	A list with all the reason codes found upon search criteria.

Resolve Call Get Next Response

Table 328 – Resolve Call Get Next Response

Type	Name	Description
StartOrderFlow	NextStep	The next valid step in the Start order process
OpenCallDTO	OpenCall	An object containing the Call details

Get Allowed Brands Open Calls Response

Table 329 – Get Allowed Brands Open Calls Response

Type	Name	Description
List<BrandDisplayItemDTO>	BrandList	A list containing dtos of the available brands
Int	BrandId	The current brand id in session for the user
Int	NumberOfBrands	The number of brands available for the user
Int	NumberOfOpenCalls	The number of open calls for the user
StartOrderFlow	NextStep	The next valid step in the Start order process
OpenCallDTO	OpenCall	An object containing the Call details

Login User Start Session Response

Table 330 – Login User Start Session Response

Type	Name	Description
LoginUserStartSessionResultType	ResultType	A response status enumeration.
SessionDTO	Session	An object used to represent the customer's session

Get User Account Response

Table 331 – Get User Account Response

Type	Name	Description
UserAccountDTO	UserAccount	An object to represent customer's account information

Add Address Response

Table 332 – Add Address Response

Type	Name	Description
AddAddressResponseStatus	ResponseStatus	A response status enumeration.
AddressDTO	CustomerAddress	An object used to represent the customer's location.

Apply Voucher Response

Table 333 – Apply Voucher Response

Type	Name	Description
String	VoucherCode	Applied voucher code.
String	VoucherDescription	The description of returned voucher
Bool	IsVoucherValid	Flag indicating whether voucher code supplied is valid.
Bool	VouchersEnabled	Flag indicating whether vouchers are enabled.
Bool	VoucherActive	Flag indicating whether voucher is active.

String	TermsAndConditionsText	Terms and conditions for voucher
String	TermsAndConditionsUrl	Url pointing to terms and conditions for voucher
String	DisplayMessages	Validation messages for vouchers.
VoucherDiscountStatusCode	VoucherQualifierStatusCode	Voucher Qualify Status Code
ProviderBasketItemDTO	AutomaticAddedProduct	The offer product that got added automatically to the basket when voucher is applied
ProviderBasketItemDTO	UpsellAddedProduct	The offer product that got added automatically to the basket when upsell is applied
VoucherValidationDetailsDTO	VoucherValidationDetails	Contains information about conditions that users need in order to avail offer

Get Basket Response

Table 334 – Get Basket Response

Type	Name	Description
BasketDTO	Basket	An object used to represent the current customer order basket.
BasketResponseStatus	ResponseStatus	A response status enumeration.
List<ProductDTO>	ProductsNotAdded	List of products not added to the basket following an add method.
Bool	VoucherQualifies	True if a voucher has been applied and the voucher qualifies for the offer
Bool	IsVoucherApplied	True if a voucher has been applied
	VoucherErrorMessage	

List<OfferProductDTO>	VoucherProducts	A list of products that are to be offered to the customer as part of an applied voucher. These are in hierarchical format. At the top level is a list of groups, each group contains children, each child being an offer product.
ProviderBasketItemDTO	AutomaticAddedProduct	The offer product that got added automatically to the basket
List<OfferProductDTO>	UpSellProducts	A list of products that are to be offered to the customer as part of an UpSell rule. These are in hierarchical format. At the top level is a list of groups, each group contains children, each child being an offer product.
ProviderBasketItemDTO	UpsellAddedProduct	The offer product that needed configuration which was added through Up-Sell

Get Bucket Configurator Response

Table 335 – Get Bucket Configurator Response

Type	Name	Description
BucketRootProductDto	RootBucketProduct	An object which represents the root product of the bucket product. This object contains details of the root product and configuration state.

Update Bucket Item Response

Table 336 – Update Bucket Item Response

Type	Name	Description
ContentDataDTO	ContentData	An object used to represent Content.
GetContentResultType	ResultType	A response result enumeration

Get Content Response

Table 337 – Get Content Response

Type	Name	Description
ContentDataDTO	ContentData	An object used to represent Content.
GetContentResultType	ResultType	A response result enumeration

Get Customer Account Response

Table 338 – Get Customer Account Response

Type	Name	Description
AddressDTO[]	CustomerAddresses	A list of objects used to represent customer locations.
PaymentStoredCardDTO[]	CustomerPaymentCards	A list of objects used to represent customer payment methods.
CustomerDTO	CustomerDetails	An object used to represent a customer’s details.
GetCustomerAccountResponseStatus	ResponseStatus	A response status enumeration

Get Menu And Start Order Response

Table 339 – Get Menu And Start Order Response

Type	Name	Description
CategoryDTO	MenuRootCategory	The root of the menu hierarchy. This consists of a category that contains a list of products and a list of sub-categories. This is the entire menu available at the selected store.
int	MenuId	The menu Identifier associated with the menu data returned.
CurrencyDTO	Currency	The currency associated with the current store.
StoreStartOrderResultType	ResultType	A response status enumeration.
List<ProductTagDTO>	ProductTagLegend	

Get Order Confirmation Response

Table 340 – Get Order Confirmation Response

Type	Name	Description
OrderDTO	Order	The order details.
StoreDTO	Store	The store details where order has been placed.
String	CustomerEmailAddress	Customer email address.
String	CustomerPhoneNumber	Customer telephone number.
String	CustomerFirstName	The first name of the customer.
String	CustomerLastName	The last name of the customer.
Bool	OrderSuccess	The flag indicating if order has been placed successfully.
String	OrderStatusMessage	Status message for the order.
Int	CustomerId	The customer identifier.

Get Store Order Options Response

Table 341 – Get Store Order Options Response

Type	Name	Description
List<StoreOrderOptionDTO>	StoreOrderOptions	A list of options (time slots and order classes) available for order placement for the currently selected store and order date.
GetStoreOrderOptionsResult Type	ResultType	A response type enumeration.
DateTime	StoreLocalisedDateTime	Current store local date time.
double	StorePromiseTimeSpan	Delivery promise time for store.
string	StoreMessage	The Store message.
Bool	CollectNowAvailable	The flag indicates whether Collection is available now or not
Bool	DeliverNowAvailable	The flag indicates whether Delivery is available now or not
DateTime	CollectNowPromiseTime	PromiseTime for Collection
DateTime	DeliverNowPromiseTime	PromiseTime for Delivery
TimeSpan	CollectNowPromiseTime Span	Collection PromiseTime in Minutes
TimeSpan	DeliverNowPromiseTime Span	DeliveryPromiseTime in Minutes
DateTime	BusinessDayDate	Store Business Time

Store Attribute Response

Table 342 – Store Attribute Response

Type	Name	Description
List<StoreAttributeDTO>	StoreOrderOptions	A list of options (time slots and order classes) available for order placement for the currently selected store and order date.

Login Response

Table 343 – Login Response

Type	Name	Description
LoginResultType	ResultType	A response type enumeration.
UrlEncodedSessionToken	String	Session token for newly created session.

Password Recovery Response

Table 344 – Password Recovery Response

Type	Name	Description
PasswordRecoveryResponseStatus	ResponseStatus	A response status enumeration.

Place Order Response

Table 345 – Place Order Response

Type	Name	Description
PlaceOrderStatus	OrderStatus	A response status enumeration.
DateTime	OrderPromiseTime	Order promise date time.
String	OrderConfirmationNumber	Order confirmation number.
PaymentMoreInfoRequired DTO	MoreInforRequiredRequest	More information required while placing order.
String	LastOrderStorePhoneNumber	The phone number of store where last order is placed.
String	POSValue	The value that returned by POS provider and it can be used to process for any purpose needed in UISS. Currently, It is used to return the QR code for POS QR.

Register Response

Table 346 – Register Response

Type	Name	Description
RegisterResponseStatus	ResponseStatus	A response status enumeration.

Remove Address Response

Table 347 – Remove Address Response

Type	Name	Description
RemoveAddressResponseStatus	ResponseStatus	A response status enumeration.

Start Session Response

Table 348 – Start Session Response

Type	Name	Description
SessionDTO	Session	A session object which is to be retained on the calling application for future requests to all service methods for the same customer.
StartSessionResultType	ResultType	A response status enumeration.

Store List Response

Table 349 – Store List Response

Type	Name	Description
StoreDTO[]	StoreList	A list of store objects that will accept orders for the customer based on their location.
StoreSearchResultType	ResultType	A response status enumeration.
int	TotalPageCount	Total number of pages for a given page size.
int	TotalStoreCount	Total number of stores

Store Search Response

Table 350 – Store Search Response

Type	Name	Description
StoreDTO[]	StoreList	A list of store objects that will accept orders for the customer based on their location.
StoreSearchResultType	ResultType	A response status enumeration.

Store Detail Response

Table 351 – Store Detail Response

Type	Name	Description
StoreDTO	Store	The detailed store object.

Store Names Response

Table 352 – Store Names Response

Type	Name	Description
StoreLiteDTO[]	Stores	A list of lightweight store objects

Update Address Response

Table 353 – Update Address Response

Type	Name	Description
UpdateAddressResponseStatus	ResponseStatus	A response status enumeration.
AddressDTO	CustomerAddress	An object used to represent the customers location.

Update Customer Response

Table 354 – Update Customer Response

Type	Name	Description

Address Search Response

Table 355 – Address Search Response

Type	Name	Description
AddressDTO[]	Addresses	An array of address objects containing the search results
int	ResultSize	Represents the overall size of the result of the query. This number is the overall size whether a paged response is required or not.

Get Customer Orders Response

Table 356 – Get Customer Orders Response

Type	Name	Description
WishListDTO[]	WishLists	An array of WishList objects
RecentOrder[]	RecentOrders	An array of RecentOrder objects.
int	TotalRecentOrderCount	Total number of customer recent order count.
Bool	AllowOrderPlacement	Flag indicating whether user is allowed to place order.

Save Basket As Wishlist Response

Table 357 – Save Basket As Wishlist Response

Type	Name	Description

Get Configurator Response

Table 358 – Get Configurator Response

Type	Name	Description
ProductConfigurationStateDTO	ConfigurationState	An object containing the current configuration of the selected product.
ProductConfigurationOptionsDTO	ConfigurationOptions	An object containing the current available configuration selections.
List<ProductConfigurationStateDTO>	SectionConfiguration	Contains a list of configuration states and options for multiple sections.
ProviderBasketItemDTO	AutomaticAddedProduct	The offer product that got added automatically to the basket
ProviderBasketItemDTO	UpsellAddedProduct	The offer product that needed configuration which was added through Up-Sell

Configure Product Response

Table 359 – Configure Product Response

Type	Name	Description
ProductConfigurationActions[]	SuccessfulActions	List of actions performed successfully.
ProductConfigurationValidation	ValidationOutcome	Result of product configuration validation.
ConfigurationDTO	ConfigurationState	Current state of complex product configuration.

Deal Completed Response – Inherits Get Basket Response

Table 360 – Deal Completed Response – Inherits Get Basket Response

Type	Name	Description

Get Deal Configurator Response

Table 361 – Get Deal Configurator Response

Type	Name	Description
DealConfigurationDTO	DealSteps	Contains list of deal steps configuration.
ProductConfiguratorKeyValuePairDTO[]	SelectedProducts	Contains current selection of product in a deal if editing deal, it would be empty if adding a deal.

Get Payment Options Response

Table 362 – Get Payment Options Response

Type	Name	Description
PaymentMethodType[]	PaymentMethodsAvailable	An array of payment methods. Methods include items such as Cash or CreditDebitCard.
PaymentCardOptionDTO[]	PaymentCardOptions	An array of payment card options tying up the card type and payment network provider.
PaymentStoredCardDTO[]	CustomerStoredCards	An array of stored payment cards for the customer.

Get Checkout Payment Options Response

Table 363 – Get Checkout Payment Options Response

Type	Name	Description
PaymentMethodType[]	PaymentMethodsAvailable	An array of payment methods. Methods include items such as Cash or CreditDebitCard.
PaymentCardOptionDTO[]	PaymentCardOptions	An array of payment card options tying up the card type and payment network provider.
PaymentStoredCardDTO[]	CustomerStoredCards	An array of stored payment cards for the customer.

Place Order Complete Payment Response

Table 364 – Place Order Complete Payment Response

Type	Name	Description
Decimal	TotalAmountOfTax	The total amount of tax
Decimal	TotalPrice	The order total
Decimal	TotalPriceOriginal	The order total before taxes and surcharges
Decimal	TotalAmountOfDiscount	Total discount
Decimal	TotalAmountOfDiscount	Discount amount if there are any
String	POSResponse	Response string from POS provider

Calculate Total Response

Table 365 – Calculate Total Response

Type	Name	Description
PlaceOrderStatus	OrderStatus	The status of order placement.
DateTime	OrderPromiseTime	The Order promise time
String	OrderConfirmationNumber	The order confirmation number.

Place Order Response

Table 366 – Place Order Response

Type	Name	Description
PlaceOrderStatus	OrderStatus	The status of the order, e.g. Success, NoValidSlotsFound.
DateTime	OrderPromiseTime	The promise time for the order.

Activate Loyalty Account Response

Table 367 – Activate Loyalty Account Response

Type	Name	Description
LoyaltyAccountSummaryDTO	AccountSummary	Details of the loyalty account specified in the request.

Check Loyalty Account Balance Response

Table 368 – Check Loyalty Account Balance Response

Type	Name	Description
LoyaltyAccountSummaryDTO	AccountSummary	Details of the loyalty account specified in the request.

Create Loyalty Account Response

Table 369 – Create Loyalty Account Response

Type	Name	Description
LoyaltyAccountSummaryDTO	AccountSummary	Details of the loyalty account specified in the request.

Credit Loyalty Account Response

Table 370 – Create Loyalty Account Response

Type	Name	Description
KeyValuePairDTO[]	Attributes	List of key value pairs representing additional information.

Lost Loyalty Card Response

Table 371 – Lost Loyalty Card Response

Type	Name	Description
KeyValuePairDTO[]	Attributes	List of key value pairs representing additional information.

Get Loyalty Account Transaction History Response

Table 372 – Get Loyalty Account Transaction History Response

Type	Name	Description
LoyaltyTransactionSummaryDTO[]	TransactionList	A list of transaction detail objects.

Get Loyalty Coupon List Response

Table 373 – Get Loyalty Coupon List Response

Type	Name	Description
LoyaltyCouponDTO[]	CouponList	A list of coupon details objects.

List Loyalty Accounts Response

Table 374 – List Loyalty Accounts Response

Type	Name	Description
LoyaltyAccountSummaryDTO[]	AccountSummary	A list of account summary details.

Create Loyalty Account Response

Table 375 – Create Loyalty Account Response

Type	Name	Description
boolean	AccountIsValid	Boolean value where true means the account is valid and false means the account is NOT valid.

Update Loyalty Account Response

Table 376 – Update Loyalty Account Response

Type	Name	Description
KeyValuePairDTO[]	Attributes	List of key value pairs representing additional information.

Link Existing Account Response

Table 377 – Link Existing Account Response

Type	Name	Description
KeyValuePairDTO[]	Attributes	List of key value pairs representing additional information.

Get Unique Items Response

Table 378 – Get Unique Items Response

Type	Name	Description
KeyValuePairDTO[]	ItemList	List of key value pairs representing additional information.

Apply Voucher Response

Table 379 – Apply Voucher Response

Type	Name	Description
String	VoucherCode	The code of the voucher that has been applied to the basket
String	**VoucherDescription**	A description of the voucher
Bool	IsVoucherValid	True if the voucher is valid and was applied to the basket, false otherwise
String	TermsAndConditionsText	Text describing and terms and conditions that apply to the voucher
string	TermsAndConditionsURL	A URL of a web page that contains terms and conditions that apply to the voucher
String[]	DisplayMessages	Any messages that are returned, usually an explanation of why the voucher was not applied
bool	VouchersEnabled	Reserve for future expansion
Bool	VoucherActive	Reserved for future expansion

Get Basket Apply Voucher Response

Table 380 – Get Basket Apply Voucher Response

Type	Name	Description
	Basket properties	Inherits from GetBasketResponse and therefore contains all of its properties
String	VoucherCode	The code of the voucher that has been applied to the basket
String	VoucherDescription	A description of the voucher
Bool	IsVoucherValid	True if the voucher is valid and was applied to the basket, false otherwise
bool	VouchersEnabled	Is voucher enabled for the current context
Bool	VoucherActive	Is voucher active for the current context
String	TermsAndConditionsText	Text describing and terms and conditions that apply to the voucher
string	TermsAndConditionsURL	A URL of a web page that contains terms and conditions that apply to the voucher
String[]	DisplayMessages	Any messages that are returned, usually an explanation of why the voucher was not applied

Remove Voucher Response

Table 381 – Remove Voucher Response

Type	Name	Description
boolean	AccountIsValid	Boolean value where true means the account is valid and false means the account is NOT valid.

Get Basket Remove Voucher Response

Table 382 – Get Basket Remove Voucher Response

Type	Name	Description
	Basket properties	Inherits from GetBasketResponse and therefore contains all of its properties

Validate Basket Against Response

Table 383 – Validate Basket Against Response

Type	Name	Description
	Basket properties	Inherits from GetBasketResponse and therefore contains all of its properties
Bool	IsReApplyVoucher	Flag indicate whether the applied voucher is reapplied to refreshed basket
Bool	IsReApplyVoucherSucess	Flag indicate whether the applied voucher is successfully reapplied to refreshed basket
String	VoucherCode	Voucher code to apply
String	VoucherDescription	Description of the voucher
Bool	IsVoucherValid	True if voucher code is valid
Bool	VouchersEnabled	True if voucher is enabled
Bool	VoucherActive	True if voucher is active
String	TermsAndConditionsText	Terms and conditions for the voucher
String	TermsAndConditionsUrl	Url that contains terms and conditions for the voucher
String[]	DisplayMessages	The validation result message
Bool	PersistStateForRevertChanges	Flag to enable FrontEnd recognize if RevertValidateBasetAgainstChanges MUST be called True: FrontEnd MUST call RevertValidateBasetAgainChanges immediately after ValidateBasketAgainstChanges finished
StoreStartOrderResultType	ResultType	Response status enumeration
DateTime	DateOrderRequired	New Order Time
Int	StoreId	Current StoreId after reverting changes

Confirm Validate Basket Against Changes Response

Table 384 – Confirm Validate Basket Against Changes Response

Type	Name	Description
DateTime	DateOrderRequired	Valid Order Time after confirm or revert changes
Int	StoreId	Current in-used Store Id after confirm or revert changes

Add Invitee Response

Table 385 – Add Invitee Response

Type	Name	Description
InviteeDTO	GroupOrderInvitee	An object used to represent the Invitee.

Get Invitee Response

Table 386 – Get Invitee Response

Type	Name	Description
List<InviteeDTO>	GroupOrderInvitee	List of Invitees that are associated with the customer.

Get Grouporder Add Order Response

Table 387 – Get Grouporder Add Order Response

Type	Name	Description
GroupOrderDTO	GroupOrder	An object used to represent the GroupOrder.
String	OrderStatus	Status of the grouporder
Decimal	OrderTotal	OrderTotal for the whole event
Decimal	OrderTax	OrderTax for the whole event

Get Limit Response

Table 388 – Get Limit Response

Type	Name	Description
GroupOrderInviteeInstanceDTO	GroupOrderInviteeInstance	An object used to represent the GroupOrderInviteeInstance.

Get Grouporder By Customerid Response

Table 389 – Get Grouporder By Customerid Response

Type	Name	Description
List<GroupOrderDTO>	GroupOrderList	List of grouporder objects created by the customer.

Add Inviteegroup Response

Table 390 – Add Inviteegroup Response

Type	Name	Description
InviteeGroupDTO	InviteeGroup	An object used to represent the InviteeGroup.

Add Invitee to Inviteegroup Response

Table 391 – Add Invitee to Inviteegroup Response

Type	Name	Description
List<InviteeDTO>	InviteeList	List of Invitee objects associated with the invitee group

Reset Session Order Details Response

Table 392 – Reset Session Order Details Response

Type	Name	Description

Save Product Review Response

Table 393 – Save Product Review Response

Type	Name	Description
String	ErrorMessage	Error message if rating is invalid
Decimal	Average	Average rating of the product
Int	NumberOfRatings	Total number of ratings received for this product.

Find Alternative Products Response

Table 394 – Find Alternative Products Response

Type	Name	Description
List<ProductDTO>	Products	List of alternative products

Get Product By Product Id Response

Table 395 – Get Product By Product Id Response

Type	Name	Description
ProductDTO	Product	The product

Get Product By Category Response

Table 396 – Get Product By Category Response

Type	Name	Description
CategoryDTO	Category	The Category

User Account Status Response

Table 397 – User Account Status Response

Type	Name	Description
UserAccountStatusResultType	ResultType	Type of user account status.

Forgot Password Response

Table 398 – Forgot Password Response

Type	Name	Description
ForgotPasswordResponseStatus	ResponseStatus	Status of forgot password request whether it successfully processed and user is allowed to recover forgotten password.

Reset Password Response

Table 399 – Reset Password Response

Type	Name	Description
ResetPasswordResponse	ResponseStatus	Status of reset password request whether it successfully processed.
String	UserEmailAddress	Email address of user

Address Validation Response

Table 400 – Address Validation Response

Type	Name	Description
Bool	IsValid	Flag indicating whether address is valid.
List<AddressDTO>	Address	List of matching address
Int	ResultSize	Number of total results, contains a total number of addresses from all pages.
List<KeyValuePair<string, string>>	AddressValidationkey Values	List of key-value pairs of validation errors.

Get Address Type Ahead TownCity By Region Response

Table 401 – Get Address Type Ahead TownCity By Region Response

Type	Name	Description
AddressDTO[]	Addresses	An array of address objects containing the search results
int	ResultSize	Represents the overall size of the result of the query. This number is the overall size whether a paged response is required or not.
List<KeyValuePair<string, string>>	AddressValidationkey Values	List of key-value pairs of validation errors.

Get Address Type Ahead Street By TownCity Response

Table 402 – Get Address Type Ahead Street By TownCity Response

Type	Name	Description
AddressDTO[]	Addresses	An array of address objects containing the search results
int	ResultSize	Represents the overall size of the result of the query. This number is the overall size whether a paged response is required or not.
List<KeyValuePair<string, string>>	AddressValidationkey Values	List of key-value pairs of validation errors.

Get Address Type Ahead District By TownCity Response

Table 403 – Get Address Type Ahead District By TownCity Response

Type	Name	Description
AddressDTO[]	Addresses	An array of address objects containing the search results
int	ResultSize	Represents the overall size of the result of the query. This number is the overall size whether a paged response is required or not.
List<KeyValuePair<string, string>>	AddressValidationkey Values	List of key-value pairs of validation errors.

Get Address Type Ahead Region Response

Table 404 – Get Address Type Ahead Region Response

Type	Name	Description
AddressDTO[]	Addresses	An array of address objects containing the search results
int	ResultSize	Represents the overall size of the result of the query. This number is the overall size whether a paged response is required or not.
List<KeyValuePair<string, string>>	AddressValidationkey Values	List of key-value pairs of validation errors.

Get Address Type Ahead Addresses Response

Table 405 – Get Address Type Ahead Addresses Response

Type	Name	Description
String[]	Addresses	An array of string values containing comma separated values of address fields specified in SearchReturnAddressFields property of request object.

Authenticate Employee Response

Table 406 – Authenticate Employee Response

Type	Name	Description
String	FirstName	First name of employee
String	LastName	Last name of employee
String	DisplayName	Display name of employee
Long	EmployeeID	Employee identifier.

SignOut Employee Response

Table 407 – SignOut Employee Response

Type	Name	Description

Get Client Configuration Response

Table 408 – Get Client Configuration Response

Type	Name	Description
String	Applicationid	Client application identifier
String	DefaultCultureCode	Default culture for client application
List<KeyValuePairDTO>	Attributes	List of application setting configuration key-value pairs

Get Client Locale Configuration Response

Table 409 – Get Client Locale Configuration Response

Type	Name	Description
List<ClientCultureDTO>	ClientCultureList	List of supported client cultures

Get Deal Configuration By Key Response

Table 410 – Get Deal Configuration By Key Response

Type	Name	Description
String	ImageUrl	Display image url for the deal
Int	ProductId	Product identifier for the deal in OHEICS

Get Configurations Response

Table 411 – Get Configurations Response

Type	Name	Description
String	Applicationid	Client application identifier
String	DefaultCultureCode	Default culture for client application
List<KeyValuePairDTO>	Attributes	List of application setting configuration key-value pairs

Get LocalisationConfigurations Response

Table 412 – Get LocalisationConfigurations Response

Type	Name	Description
GetLocalisationConfigurations ResponseStatus	ResponseStatus	The response status indicating whether the requested configurations were found or not.
List<LocalisationConfiguration DTO>	Configurations	List of localization Configurations

Map Localization Configuration To Address Response

Table 413 – Map Localization Configuration To Address Response

Type	Name	Description
AddressDTO	Address	The mapped address
Bool	IsValid	Flag indicating whether address mapping is valid or not
List<KeyValuePair<string, string>>	MappingValidation Messages	Mapping validation messages if validation fails for any of the element.

Map Address To Localization Configuration Response

Table 414 – Map Address To Localization Configuration Response

Type	Name	Description
LocalisationConfigurationDTO	Configuration	Mapped localization configuration

Get Default Balance Response

Table 415 – Get Default Balance Response

Type	Name	Description
Decimal	Amount	Balance amount
Bool	HasGiftCard	Flag indicating whether customer has gift card or not
String	FirstName	First name of the customer on the default gift card
String	LastName	Last name of the customer on the default gift card

Add Customer Paymentcard Response

Table 416 – Add Customer Paymentcard Response

Type	Name	Description
PaymentStoredCardDTO	CustomerPaymentCard	The customer payment card with its Id

Update Customer Paymentcard Response

Table 417 – Update Customer Paymentcard Response

Type	Name	Description
PaymentStoredCardDTO	CustomerPaymentCard	The customer payment card with its Id

Get All Paymentcards Response

Table 418 – Get All Paymentcards Response

Type	Name	Description
List<PaymentCardDTO>	PaymentCards	All payment card types.

Hide Past Order Response

Table 419 – Hide Past Order Response

Type	Name	Description
HidePastOrderResultType	HidePastOrderResult	Status of the process hiding Past Order

Data Transfer Objects

Basket Discount Type DTO

Provides information on discount items.

Table 420 – Basket Discount Type DTO

Type	Name	Description
int	BasketDiscountTypeId	The basket discount unique identifier
int	DiscountTypeId	The discount type identifier
string	DiscountTypeText	The discount type description
bool	IsDefaultDiscountAmountPercentage	True if discount is percentage, false if it's amount
decimal	DiscountAmount	The discount amount (or percentage)

Brand DTO

Provides information on brand items.

Table 421 – Brand DTO

Type	Name	Description
int	BrandId	The brand item identifier
string	DisplayTitleText	The brand item description

Channel DTO

Provides information on channel items.

Table 422 – Channel DTO

Type	Name	Description
int	Channeld	The channel item identifier
string	DisplayTitleText	The channel item description
int	BrandId	The id of the brand to which the channel belongs to.

User Type DTO

Provides information on user types.

Table 423 – User Type DTO

Type	Name	Description
int	UserTypeId	The user type unique identifier
string	UserTypeDescription	The user type description

Login Period DTO

Provides information on login periods.

Table 424 – Login Period DTO

Type	Name	Description
int	LoginPeriodId	The login period unique identifier
String	LoginPeriodDescription	The login period description

Region DTO

Provides information on regions.

Table 425 – Region DTO

Type	Name	Description
int	RegionId	The region unique identifier
String	RegionDescription	The region description

Time Zone Info DTO

Provides information on time zones.

Table 426 – Time Zone Info DTO

Type	Name	Description
int	Id	The time zone identifier
String	DisplayName	The time zone name

Customer Detail DTO

Provides information on customers (relevant in a call center context).

Table 427 – Customer Detail DTO

Type	Name	Description
CustomerDTO	Customer	The customer object
OrderDTO	LastOrder	The customer’s last order
CustomerNoteDTO	LatestComplaint	Customer’s latest complaint
decimal	AverageCustomerOrderValue	The reason code for the pending discount
decimal	ComplaintCount	Number of complaints from this customer.
Datetime	PendingDiscountIssuedOn	The date the pending discount was issued
string	PendingDiscountIssuedBy	The operator who issued the discount
string	PendingDiscountType	The discount type
string	PendingDiscountReason	The discount reason code

Customer Result DTO

Provides information on customers.

Table 428 – Customer Result DTO

Type	Name	Description
CustomerDTO	Customer	The customer object
AddressDTO	CustomerAddress	The customer’s address object
bool	IsDiscountRequired	A flag indicating if the customer has a pending discount
string	DiscountReason	The reason code for the pending discount
DateTime	LastOrder	Customer’s last order date.

Customer Note Type DTO

Provides information on customer notes type items.

Table 429 – Customer Note Type DTO

Type	Name	Description
int	CustomerNoteTypeId	Customer note type unique identifier
CustomerNoteTypeClass	CustomerNoteTypeClass	The customer note type class
string	DisplayTitleText	The customer note type description

Customer Note DTO

Provides information on customer notes items.

Table 430 – Customer Note DTO

Type	Name	Description
int	CustomerNoteId	Customer note unique identifier
int	CustomerId	The customer id
int	TypeId	Customer note type id
string	TypeText	Customer note description
int	ClassId	Customer note class id
string	ClassText	Customer note class description
Int	ReasonCodeId	The reason code id. This is mainly used for complaint type customer notes, in order to set a resolution (i.e. grant a discount)
String	ReasonCodeText	The reason code description.
Int	ParentId	The parent customer note identifier (when multiple resolutions have been created for a given customer note, this property will usually contain the id of the latest resolution

		customer note created or in the case where there is only one resolution, this property will contain the id of the original complaint type customer note)
Int	RootParentId	The top-level customer note identifier (normally this will always be the id of the original complaint type customer note
Int	OrderId	The order id associated to the customer note item (for order complaint type customer notes)
string	NoteText	Customer note comments
bool	IsProcessed	Indicates if the customer note has been resolved (usually this is used for complaint type customer notes, to indicate if the complaint has been resolved either with a discount, a callback or any other kind of resolution)
string	UpdatedByName	Operator name
string	OrderReference	The reference identifier of the associated order.
Datetime	DateCreated	The date the customer note was created
Datetime	DateUpdated	The date the customer note was last updated
string	TimeZoneForDateCreated	Indicates the timezone of the DateCreated property
List<CustomerNote>	ChildNotes	Child customer notes

POS Order Summary DTO

Provides information on open call items.

Table 431 – POS Order Summary DTO

Type	Name	Description
long	OderNumber	Unique order number
long	OrderSequence	Order sequence
string	OrderType	Order type
string	OrderTotal	Order total amount
DateTime	CreatedTime	The date the order was created
Datetime	LastServiceTime	The date the order was last serviced
bool	IsFutureOrder	Flag to indicate whether or not it's a future order
bool	IsDelayedOrder	Flag to indicate whether or not it's a delayed order
Datetime	AutoFireTime	The auto-fire date (when the order is sent to Kitchen Displays)

Call Order Compact DTO

A simplified version of Call Order DTO.

Table 432 – Call Order Compact DTO

Type	Name	Description
int	CustomerId	A unique identifier for the customer
int	OrderId	The order item identifier
string	OrderReference	The order item reference
string	Channel	The channel item description
string	FirstName	Customer's first name.
string	LastName	Customer's last name.
String	DeliveryAddress	The delivery address
String	Phone	Customer's phone number
Datetime	OrderDateCreated	The date by when order was created
Datetime	OrderDateRequired	The date by when order is to be required.
String	Store	The Store name

String	PosResponse	The response received back from the POS when the order was placed
String	PosConfirmationNumber	The order confirmation number generated in the POS (i.e. Check # + Check Seq)
String	OrderType	The order type
String	TakenBy	The name of the user who took the order (operator)
String	StatusClass	The order status class description
String	CallbackReasonCode	The callback reason code (applies only for callback orders)
bool	IsApprovalAllowed	Flag to indicate whether or not the order qualifies for Approval (applies only to callback order)

Call Order DTO

Call Order DTO is an extension of OrderDTO, this data object includes properties that are relevant only in a call center context.

Table 433 – Call Order DTO

Type	Name	Description
int	CustomerId	A unique identifier for the customer
string	OrderStatusNote	Order status note, i.e. information for callback approved or rejected
string	OrderStatusOperator	The user who updated the order status most recently.
string	Channel	The channel item description
string	FirstName	Customer’s first name.
string	LastName	Customer’s last name.
String	DeliveryAddress	The delivery address
String	Phone	Customer’s phone number
string	OrderDateCreated	The date by when order was created
String	OrderDateRequired	The date by when order is to be required.
string	DeliveryNotes	Oder delivery notes
String	PosResponse	The response received back from the POS when the order was placed

String	PosConfirmationNumber	The order confirmation number generated in the POS (i.e. Check # + Check Seq)
bool	CancellationAllowed	Flag to indicate whether or not this order qualifies for Cancellation
string	CancellationComments	The order cancellation comments
String	StatusClass	The order status class description
string	ElapsedTime	Elapsed time between the current time and the time the order is required
bool	IsApprovalAllowed	Flag to indicate whether or not the order qualifies for Approval (applies only to callback order)

Reason Code DTO

Provides information on the reason code items.

Table 434 – Reason Code DTO

Type	Name	Description
int	ReasonCodeId	A unique identifier for the reason code item
ReasonCodeType	ReasonCodeType	The reason code type
string	DisplayTitleText	A description of the reason code item

OPEN CALL DTO

Provides information on open call items.

Table 435 – OPEN CALL DTO

Type	Name	Description
int	CallId	A unique identifier for the call item
string	StartTime	The call start time
string	Status	A description of the current call status
string	CustomerDetails	Call's customer information (name)

Brand Display Item DTO

Provides information on the brand items.

Table 436 – Brand Display Item DTO

Type	Name	Description
int	BrandId	A unique identifier for the brand item
string	DisplayTitleText	Brand's title
string	ImageUrl	Relative path to the Brand's logo image.

User Account DTO

The customer's account information.

Table 437 – User Account DTO

Type	Name	Description
string	ProviderId	A unique identifier the user
string	LoginName	The user login name
Title	Title	User title (enum)
String	ForeName	First name
String	LastName	Last name
Int	UserTypeId	Type of user (see GetUserAccountConfigurations method in ConfigurationService)
Int	RegionId	User region id
String	ContactTelephonePrimary	Primary telephone number
String	ContactTelephoneSecondary	Secondary telephone number
String	ContactMobile	Mobile telephone number
String	ContactEmailPrimary	Primary email
Int	LoginPeriodId	Type of login period (see GetUserAccountConfigurations method in ConfigurationService)
int	LoginStartTimeHour	Login start hour
Int	LoginStartTimeMinute	Login start minute
Int	LoginEndTimeHour	Login end hour
Int	LoginEndTimeMinute	Login end minute
Datetime	DateLastLogin	The date of the last login
Datetime	DateCreated	Date of creation

Datetime	DateUpdated	Date of the last modification
Bool	IsActive	Flag to indicate if the user account is active
string	TimeZoneInfoId	User timezone

AddressDTO

The Address DTO is a generic container for location information. This object is used for both store addresses and customer addresses.

Table 438 – AddressDTO

Type	Name	Description
string	AddressId	An address identifier.
string	BuildingLetter	The building letter (5 characters max).
string	BuildingName	Building/house name if any.
int	BuildingNumber	Building/house number.
string	District	The district or suburb – this is usually an area within a town/city.
decimal	Latitude	The locations latitude
decimal	Longitude	The locations longitude
string	OrganisationName	The name of the organization if any.
string	PostCodeOrZip	The address postal code or zip code
string	StreetName	The street name for the address.
string	Territory	The territory, state or county.
string	TownCity	The town or city.
int	CountryId	A unique identifier for the country for the address.
string	CountryText	The name of the country for the address.
string	Intersection	The intersection street
string	AddressDescription	Unformatted address string
string	ProviderAddressKey	Provider specific address key. i.e. for the postcode anywhere provider, this could be the InterimResult ID
AddressType	AddressType	Type of address. i.e. Residential, Business, etc.
string	RoomNumber	Room number, in case address is hotel.

string	TaxNumber	The tax number
bool	IsFavourite	Indicating whether this is favorite address.
string	BuildingNumberHigh	The building number high
string	BuildingNumberLow	The building number low
string	BuildingNumberParity	The building number parity
string	RoomNumberHigh	The room number high
string	RoomNumberLow	The room number low
string	RoomNumberParity	The room number parity
string	Landmark	Nearest landmark name
string	FloorNumber	The floor number

Basket DTO

A Basket DTO object represents a current order in progress for a customer. The basket contains items which are added and removed by the customer from available menu items. When an order is placed, the basket is converted into an order which can be sent to the point of sale system for processing.

Table 439 – Basket DTO

Type	Name	Description
Int	BasketId	A unique identifier for the basket.
List<BasketItemDTO>	BasketItems	A list of basket items.
List<BasketItemDTO>	BasketSurcharges	A list of basket surcharge items.
List<BasketTaxDTO>	BasketTaxes	A list of basket tax items.
Decimal	NetItemTotal	The net total value of the basket items in the basket.
Decimal	DiscountAmount	The total discount value applied to the basket.
decimal	SavedAmount	The saved amount is the total price different of reduced price products in the basked to the original price total of these products. This value does not include discounts.

Decimal	BasketTotal	The overall total value of the basket, including net item total, delivery charges, discounts and tax.
bool	IsCheckoutAllowed	Flag indicating whether this instance of basket is checkout allowed.
decimal	MinimumOrderValue	Minimum order value to allow checkout.
int	VoucherId	The voucher identification if voucher id applied.
string	VoucherCode	The voucher code if voucher is applied.
string	VoucherDescription	The voucher description if voucher applied.
decimal	VoucherDiscountAmount	The amount of voucher discount if voucher is applied.
bool	VoucherDiscountIsPercentage	Flag indicating if discount for applied voucher is percentage discount.
List<BasketVoucherDTO>	VoucherResult	List of all vouchers and its status in the basket.
List<BasketItemMessageDTO>	BasketItemMessages	The messages pertaining to validation and pricing of the basket items. Whether a basket item was invalid and has been removed.
DateTime	OrderTime	The time of order.
List<DiscountTypeDTO>	AvailableDiscounts[1]	The list of available discounts for the customer. The content of the list is controlled by *DiscountMode* application setting 0, All of the discounts defined in DiscountType table will be available *1,* Only the DiscountTypes applicable to the customer based on customer

		complaints resolutions will be available 2, No discount will be available
bool	IsDiscountAllowed¹	Flag to indicate if the basket instance qualifies for *Discount*.
List<DiscountTypeDTO>	AppliedDiscounts¹	The list of discounts applied to the basket
Decimal	TaxChargeAmount²	The total tax charged against the basket.
Decimal	DeliveryChargeAmount²	The total delivery charges applied to the basket.
¹These properties are only applicable to a call center context, where operators apply discounts to amend customer complaints. ²These properties are Obsolete		

Basket Item Summary DTO

The basket item summary provides information on the basket item identifier and quantity in current basket.

Table 440 – Basket Item Summary DTO

Type	Name	Description
int	BasketItemId	A unique identifier for the basket item.
int	Quantity	The quantity of the product in the basket item.

Basket Item DTO

A basket is made up of one or more basket items, each being a specific product and quantity of that product. The basket item provides information on the product, price and quantity. Once an order is placed from a basket, the basket items are converted into Order Items to be sent to the point of sale system for sale.

Table 441 – Basket Item DTO

Type	Name	Description
int	BasketItemId	A unique identifier for the basket item.
string	Description	A description of the basket item.

decimal	Price	The price of the product associated with the basket item.
int	Quantity	The quantity of the product in the basket item.
decimal	LineTotal	The total price of the basket item excluding any discount or tax (Price * Quantity). Note: This is here for backwards compatibility. Use PriceWithTaxAndDiscount.
int	ProductId	A unique identifier for the product associated with the basket item.
Boolean	IsConfigurationReqired	If true then item requires Container Model Configuration
string	ConfigurationAlias	The configuration alias.
BasketItemDTO[]	BasketItemCollection	List of related products that had been selected for this item
Bool	IsUpdateable	Is true if the server will allow the quantity to be changed. If this is false then it generally means that it's a discounted product. This setting does not affect whether the item can be deleted or not.
Decimal	UnitAmountOfDiscount	The amount of discount on a single item.
Decimal	UnitAmountOfTax	The amount of tax on a single item.
Decimal	PriceWithTaxAndDiscount	The total price of the basket item including tax and discount ((Price + UnitAmountOfTax – UnitAmountOfDiscount) * Quantity)
Decimal	UnitPrice	Unit price of basket line item.
Bool	IsVisible	Flag indicating whether product is visible.
Bool	IsDefault	Flag indicating whether line item is default.
Int	InviteeId	The invitee identifier.
String	InviteeName	The name of invitee.
Bool	IsSurcharge	Flag indicating if current line item is surcharge.

SurchargeType	SurchargeType	Surcharge type if current line item is type of surcharge.
Int	DisplayOrder	The order of display for current line item.

Basket Item Message DTO

The basket item message contains messages about specific basket items to be returned with the basket.

Table 442 – Basket Item Message DTO

Type	Name	Description
int	BasketItemId	The basket item identifier.
BasketItemMessageType	MessageType	Type of basket item message.
BasketItemMessageAction	MessageAction	The message action.
string	ProductDescription	The description of product.

Basket Tax DTO

The basket tax item provides tax information on the basket.

Table 443 – Basket Tax DTO

Type	Name	Description
decimal	Amount	The tax amount.
string	DisplayTitleTax	The display text for tax item.

Basket Product DTO

The basket product item provides product information which will be needed when adding product into the basket.

Table 444 – Basket Product DTO

Type	Name	Description
Int	Quantity	The quantity of the selected product to add to the basket. Note that this is only applicable when adding a menu product Id (See ProductType property below).

		Default quantity is 1.
Int	ProductType	BasketProductAddType enumeration value representing the type of product being added to the current basket. This will either be a menu product, a wishlist or a previous order.
Int	ProductTypeId	The Id of ProductType
Decimal	PriceOverride	Overridden price for product.
ContainerStateDTO	ContainerStateDTO	Represents related products that are added to the basket together with parent item
ContainerStateDTO	ContainerStateMissingDefaultDTO	Container state missing default DTO if it is complex product.
ContainerStateDTO	ContainerStateANPDTO	Container state ANN DTO if it is complex product.
Int	InviteeId	The invitee identifier.
String	InviteeName	The name of invitee.
String	Reference	Description when ordering product.

Category DTO

The CategoryDTO object is used to represent a container of products in the menu. This object is returned from the order service GetMenuAndStartOrder() method.

Table 445 – Category DTO

Type	Name	Description
CategoryDTO[]	Categories	A list of sub categories.
int	CategoryClassId	An identifier for the type of category.
string	CategoryClassName	The category type name.
int	CategoryId	A category identifier.
string	Description	A short description for a category.
string	ImageAltText	Image descriptive text.

string	ImageUrl	Image URL. The image can be hosted on the web server and referenced using this url.
string	MarketingDescription	A long description for a category.
ProductDTO[]	Products	A list of product objects in this category.
string	Title	The category display name.
string	LargeImageUl	Large image URL.
List<KeyValuePair<string, string>>	CategoryKeyValuePair List	List of key value pair list for category.
List<in>	ProductIds	List of poduct identifiers.
string	ItemUnavailableImage Url	ItemUnavailableImag URL.

Container State DTO

ContainerState DTO Used to represent state of selected products when modifying basked items.

Table 446 – Container State DTO

Type	Name	Description
int	ProductID	A unique identifier for the product being added.
int	Quantity	Quantity of the Product added
ContainerStateDTO[]	ProductModifiers	Related products selected with this product

Configuration Action DTO

Configuration action DTO used to represent action detail during order.

Table 447 – Configuration Action DTO

Type	Name	Description
ProductConfigurationActions	SelectedAction	An enumeration of various actions e.g. BaseSelected, SizeSelected etc.
Int	Quantity	Quantity of the Product actioned.

Int	SectionId	The identifier of the coverage (section) to apply the action to replace CoverageId property.
Int	Id	The selection identifier.
Int	ParentProductToppingGroupId	The group identifier for topping selection.
ProductToppingClass	ClassId	Class identifier for topping selection
Int	BasketId	The basket item identifier, this will be null if adding a product otherwise existing basket item identifier.

Content DataDTO

ContentData DTO represents content stored in the database.

Table 448 – Content DataDTO

Type	Name	Description
string	Name	Name of the content
long	HierarchyStrucID	Internal ID of the hierarchy on which content had been found
string	ContentType	Type of content as MIME string
Byte[]	ContentBlob	Actual content as binary blob of bytes
String	DiskFileName	File name with which this content is associated
Byte[]	LastTimeStamp	Internal stamp used to identify when content changes

Customer DTO

The Customer DTO object represents a registered customer's details.

Table 449 – Customer DTO

Type	Name	Description
int	CustomerId	A unique identifier for the customer.

string	ContactTelephonePrimary	The primary telephone number for the customer.
string	ContactTelephoneSecondary	An alternate telephone number for the customer.
string	ContactMobile	The mobile contact number of the customer.
Title	Title	An enumeration representing the customer title.
string	FirstName	The first name of the customer.
string	LastName	The last name of the customer.
string	ContactEmailPrimary	The primary email address for the customer.
string	ContactEmailSecondary	An alternate email address for the customer.
bool	MarketingOptIn	A boolean value which represents the customers desire to participate in marketing campaigns.
bool	IsBlacklisted	A boolean value to indicate whether the customer has been barred from ordering via the system.
KeyValuePairDTO[]	CustomerAttributeList	A list of key value pair objects which provide additional information about the customer.
String	ClientIPAddress	The IPAddress of the customer.
DateTime	Birthday	Birthday of Customer

Key Value Pair DTO

The Key Value Pair DTO provides a generic object which can be appended to other objects to give additional information.

Table 450 – Key Value Pair DTO

Type	Name	Description
string	Key	The key value used for indexing.
string	Value	The value associated with the key.

Opening Day Of Week DTO

The Opening Day Of Week DTO represents the store opening periods for a specific day of the week.

Table 451 – Opening Day Of Week DTO

Type	Name	Description
string	DayOfWeek	The day of the week.
OpeningTimePeriodDTO[]	OpeningPeriods	A list of objects representing opening periods within the working day.

Opening Time Period DTO

The Opening Time Period DTO represents an opening period for a store. A store can have several opening time periods during a working day, so these objects will be represented as a list of opening time periods within an Opening Day Of Week DTO object.

Table 452 – Opening Time Period DTO

Type	Name	Description
string	OpenTime	A string representation of the start of the opening period.
string	CloseTime	A string representation of the end of the opening period.

Order DTO

The Order DTO represents the order being placed by the calling application, including details of total price, discounts, delivery charges, and the items making up the order.

Table 453 – Order DTO

Type	Name	Description
int	OrderId	A unique identifier for the order.
decimal	OrderTotal	The total value of the order.
decimal	DiscountAmount	The total value of discounts on the order.
decimal	DeliveryChargeAmount	The total value of delivery charges for the order.
decimal	TaxChargeAmount	The total value of tax against the order.
String	OrderReference	The order reference.

String	FulfillmentTimeTypeText	The order fulfillment time type text.
DateTime	DateOrderRequired	The date by when order is to be required.
String	OrderMethod	The order method.
List<OrderItemDTO>	OrderItems	A list of objects representing the order items.
List<OrderTaxDTO>	OrderTaxes	A list of objects representing the order taxes.
List<OrderPaymentDTO>	OrderPayments	The list of order payment details.
String	VoucherCode	The voucher code if applied any.
String	VoucherDescription	The voucher description if applied any.
List<VoucherDTO>	VoucherResult	
String	OrderHash	The voucher OrderHash
String	StoreName	The voucher StoreName
String	StatusText	The StatusText
String	TakenBy	
String	DeliveryNotes	The DeliveryNotes
Decimal	SavedAmount	Total saved amount of Voucher and Upsell together

Order Item DTO

The Order Item DTO represents an item within an order. This object is used to define details of an order item, including the product selected, quantity, price details and additional information. Order items are sent to the point of sale with the order for processing.

Table 454 – Order Item DTO

Type	Name	Description
int	OrderItemId	A unique identifier for the order item.
int	ProductId	A unique identifier for the product.
string	ProductText	The name of the order item product.
string	ProductDescription	The description of the order item product.

int	Quantity	The quantity of the product stored in the order item.
decimal	Price	The price of each product in the order item.
decimal	PriceTotal	The total price of the order item (Price * Quantity)
bool	IsNoChargeItem	A boolean flag indicating whether this is a free order item or not.
int	DisplayOrder	A display order for the order item within the order.
DateTime	DateCreated	The date the order item was created by the user.
string	PrimarySku	A text reference to the primary standard key unit for the product for this order item.

Payment Card DTO

TBC

Table 455 – Payment Card DTO

Type	Name	Description

Order Payment DTO

The details of the payment for order. This may be required to process calculations for payment based surcharges.

Table 456 – Order Payment DTO

Type	Name	Description
Int	OrderPaymentId	The order payment identifier.
decimal	AmountTotal	The total amount paid
PaymentMethodType	PaymentMethodType	Type of payment method.

Payment DTO

The details of the payment for order. This may be required to process calculations for payment based surcharges.

Table 457 – Payment DTO

Type	Name	Description
decimal	Amount	The amount of payment.
PaymentMethodType	PaymentMethodType	Type of payment method.

Payment More Info Required DTO

Details of more information require by payment provider.

Table 458 – Payment More Info Required DTO

Type	Name	Description
List<KeyValuePairDTO>	ProviderAttributes	List of key value pair attributes for payment provider.
String	ProviderControlAlias	The provider control alias.

Payment Stored Card DTO

Details of payment cards stored against a user account for checkout.

Table 459 – Payment Stored Card DTO

Type	Name	Description
int	CustomerPaymentCardId	A unique identifier for the customer payment card.
string	CardNumberDisplayText	Display text provided by the user for the customer payment card for identification in the user interface.
PaymentCardClass	CardClass	An enumeration representing the Class of payment card.
PaymentCardType	CardType	An enumeration representing the type of payment card.

Product DTO

The product object is contains all product information.

Table 460 – Product DTO

Type	Name	Description
string	Description	A short description for a product.
string	ImageAltText	Image descriptive text.
string	ImageUrl	Image URL. The image can be hosted on the web server and referenced using this url.
string	MarketingDescription	A long description for a product.
decimal	Price	The product price.
int	ProductId	The product identifier, this is used to add a product to the basket.
int	SortOrder	The default display sort order for products in a category.
string	Title	The product name.
bool	RequiresConfiguration	A boolean flag which indicates whether this product requires custom configuration.
string	CustomerConfigurationAlias	The name of the custom configuration required for this product.
int	DisplayOrder	Display order of product
bool	IsSelected	Indicator If product is selected or not.
List<int>	ProductTagIds	List of product tag identifiers.
bool	IsPriceOverrideAllowed	Indicator id price override is allowed or not for this product.
List<ProductRepationshipDTO>	RelatedProducts	List of product relationship data transfer objects.
bool	IsContainer	Indicator if product is marked as container product.

bool	IsVisibile	Indicator if product is visible for user on user interface.
string	LargeImageUrl	Large image URL for product.
string	LeftImageUrl	The left image URL.
string	RightImageUrl	The Right image URL.
List<KeyValuePair<string, string>>	ProductKeyValuePairList	List of additional key value pair attributes for product.
ProductReviewDTO	ProductReview	The product Review

Product Base DTO – Inherits Properties of Product DTO

Table 461 – Product Base DTO – Inherits Properties of Product DTO

Type	Name	Description
int	ProductBaseTypeId	Id for pizza base type.

Order TAX DTO

The details of the tax for order.

Table 462 – Order TAX DTO

Type	Name	Description
String	TaxRateType	
Decimal	Amount	
Decimal_Payment_Method_Type	RoundedAmount	
String	DisplayAmount	

Product Configuration Options DTO

Table 463 – Product Configuration Options DTO

Type	Name	Description
ProductBaseDTO[]	BaseTypes	Available pizza base types.
ProductSizeDTO[]	Sizes	Available product sizes

ProductCoverageDTO[]	Coverages	Available product coverages e.g. whole, half and half.
ProductSpecialityDTO	Specialties	Available specialties (default configurations)
ProductToppingDTO[]	Toppings	Available toppings.

Deal Configuration DTO

Table 464 – Deal Configuration DTO

Type	Name	Description
Int	ConfigurationCurrentStep	The index of current step in deal configuration.
String	ConfigurationRootProduct Name	Name of root product of deal.
String	ConfigurationRootProduct DisplayDescription	Description of root product of deal.
String	ConfigurationRootProduct MarketingText	Marketing text of root product of deal.
DealStepConfigurationDTO[]	DealSteps	List of deal steps configuration.

Deal Step Configuration DTO

Table 465 – Deal Step Configuration DTO

Type	Name	Description
Int	ProductId	The product identifier of root product of this deal step.
String	Description	The description of root product of this deal step
Int	RootCategoryId	The category identifier of root product of this deal step.
Int	BasketitemId	The basket item identifier of current deal steps if editing basket otherwise null.
List<int>	AllowedProductIds	List of allowed product identifiers allowed for selection in current deal step.

Product Configurator Key Value Pair DTO

Table 466 – Product Configuration Key Value Pair DTO

Type	Name	Description
Int	ProductId	The product identifier of root product.
ProductConfigurationStateDTO	Configuration	Simple or container product configuration state.
ContainerStateDTO	ContainerState	Container product configuration state.
Int	BasketitemId	The basket item identifier of current deal steps if editing basket otherwise null.

Container State DTO

Table 467 – Container State DTO

Type	Name	Description
Int	ProductId	The product identifier of root product.
Int	Quantity	Quantity of product.
String	ProductKey	The product key.
Int	ParentId	The product identifier of parent product.
List<ContainerStateDTO>	ProductModifiers	List of product modifiers.

Configuration Set DTO

Table 468 – Configuration Set DTO

Type	Name	Description
Int	SectionId	Current selected section identifier.
ProductConfigurationStateDTO	ConfigurationState	Configuration state of current selected section
ProductConfigurationOptionsDTO	ConfigurationOptions	Configuration options for selected section.

Product Configuration State DTO

Table 469 – Product Configuration State DTO

Type	Name	Description
int	SelectedBaseTypeId	Selected pizza base type Id
int	SelectedSizeId	Selected product size Id
int	SelectedCoverageId	Type of coverage Id e.g. Whole, half and half etc
int	SelectedSectionId	Selected section currently being configured e.g. Whole (for whole coverage), 1st half, 2nd half.
int	SelectedSpecialityId	The specialityId for the selected product / section
ProductToppingDTO[ ]	SelectedToppings	Current toppings selected for the configured product.
decimal	CurrentPrice	Current unit cost for the selected product given current configuration.
Decimal	Price	Pass the state price back and forth as it's needed for complex products in deals.
int	Quantity	Product quantity selected.
Boolean	IsAvailable	Availability of selected product.
int	MaxToppingCount	The maximum number of toppings which can be selected for the current product.
int	MaxChangeCount	The maximum number of configuration changes allowed for this product.
int	CurrentChangeCount	The number of changes made so far while configuring the current product.
int	ProductConfigurationCl assId	The product configuration class identifier.
List<ConfigurationSet DTO>	SectionConfigurations	Contains a list of configuration states and options for multiple sections.

Product Coverage DTO – Inherits Properties of Product DTO

Table 470 – Product Coverage DTO – Inherits Properties of Product DTO

Type	Name	Description
int	ProductCoverageId	Id for product coverage.

Product Section DTO – Inherits Properties of Product DTO

Table 471 – Product Section DTO – Inherits Properties of Product DTO

Type	Name	Description
int	ProductSectionId	Id for product section.

Product Specialty DTO – Inherits Properties of Product DTO

Table 472 – Product Specialty DTO – Inherits Properties of Product DTO

Type	Name	Description
int	ProductSpecialityId	Id for product specialty.

Product Size DTO

Table 473 – Product Size DTO

Type	Name	Description
int	ProductSizeId	Id for product size.

Product Topping DTO

Table 474 – Product Topping DTO

Type	Name	Description
int	ParentProductToppingGroupId	Id for parent product topping group (part of primary key)
int	ProductToppingId	Id for product topping (part of primary key)
ProductToppingClass	ToppingClass	Class type of product topping

Product Relationship DTO – Inherits Properties of Product DTO

Table 475 – Product Relationship DTO – Inherits Properties of Product DTO

Type	Name	Description
Bool	IsDefault	Is this a default related products
Bool	IsMandatory	Is this related product mandatory for sale
Bool	IsSaleable	Can this item be sold directly
Bool	IsDiscountQualifer	Is This product used for discount qualification
Int	Quantity	How many of this product needs to be ordered

Product Review DTO

Table 476 – Product Review DTO

Type	Name	Description
decimal	Average	The average review for product.
int	NumberOfRating	The number of ratings.

Recent Order DTO

Table 477 – Recent Order DTO

Type	Name	Description
DateTime	OrderDate	Date and time of order
Int	OrderId	Order identifier
String	OrderReference	The order reference supplied to customer when this order was originally placed.
String	Moniker	The order name set by customer when this order was originally placed.
int	StoreId	Store identifier
string	StoreName	Name of the store from where user placed order.
OrderClass	OrderClass	Order class i.e. Delivery or collection
decimal	TotalOrderValue	Total value of this recent order

Long	FacebookUserId	Facebook user identifier for the user who placed order
List<RecentOrderItemDTO>	Items	List of order items
String	OrderName	Order named by customer so they can recognize each order independently in future

Recent Order Item DTO

Table 478 – Recent Order Item DTO

Type	Name	Description
Int	ProductId	Identifier of product ordered, this is assigned by my central and is read only.
String	ProductText	Display test of product
String	ProductDescription	Description of ordered product
Int	Quantity	The quantity of ordered product
Decimal	DisplayPrice	Price for line item
Bool	IsMainProduct	Flag indicating if this is main product.

Session DTO

The session DTO is created by calling the StartSession method. This object is a token used by OHEICS to represents a unique customer session, so it must be passed to all subsequent service methods.

Table 479 – Session DTO

Type	Name	Description
string	SessionId	A key used to identify the unique customer session.
string	ApplicationId	A unique key used to call the application.
string	CultureCode	The culture code that has been assigned to the customer session based on the culture code requested when the session was created. If the requested culture code is not supported the customer sill receive the default culture code for the brand.

StoreDTO

The Store DTO contains information about the restaurant or store that allows a client application to show both display information such as address, telephone number and opening times as well as available ordering options.

Table 480 – StoreDTO

Type	Name	Description
int	StoreId	The unique identifier for the store
DateTime[]	AvailableOrderSlots	The ordering times available for the same day at this store.
double	DistanceInMilesFromSeachOrigin	Distance in Miles from search origin.
string	Email	The store email address.
OpeningDayOfWeekDTO[]	OpeningHours	A list of objects representing the opening hours for the store.
AddressDTO	StoreAddress	The store address information.
string	StoreName	The store name.
OrderClass[]	SupportedOrderTypes	The available order types at this store on the current day.
String	TelephoneNumber	The store telephone number.
int	MaxFutureOrderDays	The number of days ahead orders can be placed at this store.
double	DistanceInKmsFromSearchOrigin	Distance in Kilometers from seach origin.
bool	SupportsDeliveryToSearchAddress	Indicator if store supports delivery to search address or not.
bool	IsPrimary	Indicate if store is primary or not, this flag is set for a store when searched by tradezone.
List<StoreAttributeDTO>	StoreAttributeList	List of supported store attribute. i.e. Collection, Delivery, Dine-in, Parking, etc.
CurrencyDTO	Currency	The currency associated with the current store.

Store AttributeDTO

The store attribute dto contains information about supported store attributes.

Table 481 – Store AttributeDTO

Type	Name	Description
int	StoreAttributeId	The unique identifier for store attribute.
StoreAttributeType	StoreAttributeTypeId	Store attribute type id.
string	DisplayText	Text to display on screen, This is translatable text.
string	Description	Detailed description of sore attribute.

Store Lite DTO

The store lite dto contains very basic store information, the store id and store name.

Table 482 – Store Lite DTO

Type	Name	Description
int	StoreId	The unique identifier for the store
string	StoreName	The store name.

Store Order Option DTO

The Store Order Option DTO contains information about the available start order options for the restaurant or store. A list of Store Order Option DTO objects is typically provided, one for each order class (collection, delivery etc). The DTO includes information on time slots available for ordering against.

Table 483 – Store Order Option DTO

Type	Name	Description
int	StoreId	The unique identifier for the store
List<DateTime>	AvailableOrderSlots	A list of time slots available for ordering against.
OrderClass	StoreOrderClass	The class of order that the available order slots are applicable to. I.e. collection or delivery.
List<DateTime>	AvailableOrderDateSlots	A list of date slots available for ordering against.

Validation Error DTO

A simple validation object which provides a code and description of a failure in request data validation when making a service call. Validation error DTO objects will usually be provided in a list, detailing, by code and text, which request object properties have failed validation by the service.

Table 484 – Validation Error DTO

Type	Name	Description
string	ErrorCode	A code for the validation error which can be used by the client application to consolidate validation errors with calling application objects.
string	ErrorDescription	A description of the validation rule which has not been met in this instance.

Wish List DTO

Table 485 – Wish List DTO

Type	Name	Description
int	BasketId	Basket identifier
string	BasketDescription	Basket description
int	ChannelId	Channel identifier
string	ChannelText	Channel descriptive text
datetime	ExpiryDate	Date wish list expires
int	MenuId	Menu identifier.
datetime	SavedDate	Date wish list saved
int	StoreId	Store identifier
string	StoreName	Store description
List<WishListItemDTO>	Items	List of wishlist items

Wish List Item DTO

Table 486 – Wish List Item DTO

Type	Name	Description
string	Description	Wishlist item description
int	Quantity	Quantity of wishlist item

Int	ProductId	The product identifier for product added as wishlist item
Decimal	DisplayPrice	The line total for wishlist item.

Payment Card DTO (Extends Payment DTO)

Table 487 – Payment Card DTO (Extends Payment DTO)

Type	Name	Description
int	CustomerPaymentCardId	Customer payment card ID
string	Name	Card name.
string	DisplayTitleText	Display title text
string	CardNumber	Payment card number.
string	SecurityCode	Payment card security code
string	SortCode	Payment card sort code.
int	Issue	Issue number
int	ValidYear	Payment card valid from year value.
int	ValidMonth	Payment card valid from month value.
int	ExpireYear	Payment card expiry year value.
int	ExpireMonth	Payment card expiry month value.
PaymentCardType	CardType	Payment card type e.g. Credit, Debit or Loyalty.
int	PaymentCardOptionId	Payment card option matches the card type with the payment network provider. This is the Id for the payment card option.
string	Tag	Tag
string	Pin	Pin
bool	FutureUse	Future use

Payment Stored Card DTO (Extends Payment Card DTO)

Table 488 – Payment Stored Card DTO (Extends Payment Card DTO)

Type	Name	Description

int	CustomerPaymentCardId	Stored identifier for customer payment card.
string	CardNumberDisplayText	The card number in display format (obscured)
PaymentCardType	CardType	Payment card type e.g. Credit, Debit or Loyalty.
int	PaymentCardOptionId	Payment card option matches the card type with the payment network provider. This is the Id for the payment card option.
string	Tag	Tag
string	Pin	Pin

Payment Card Option DTO

Table 489 – Payment Card Option DTO

Type	Name	Description
int	PaymentCardOptionId	Payment card option matches the card type with the payment network provider. This is the Id for the payment card option.
PaymentCardType	CardType	Payment card type e.g. Credit, Debit or Loyalty.
string	CardTitle	The Payment card title

Payment Stored Card DTO

Table 490 – Payment Stored Card DTO

Type	Name	Description
int	CustomerPaymentCardId	Customer payment card identifier.
PaymentCardType	CardType	Payment card type e.g. Credit, Debit or Loyalty.
string	CardNumberDisplayText	The display text for car number
Int	PaymentCardOptionId	Payment card option identifier

Loyalty Account Summary DTO

Table 491 – Loyalty Account Summary DTO

Type	Name	Description
string	AccountIdentifier	Loyalty account identifier.
string	Name	Loyalty account name.
string	Description	Loyalty account description.
string	Currency	ISO Currency value (i.e. GBP, USD)
int	PointsBalance	Loyalty account reward points balance.
decimal	AmountBalance	Loyalty account currency balance.
datetime	ExpiryDate	Loyalty account expiry date.
KeyValuePairDTO[]	Attributes	List of objects representing additional properties.

Loyalty Coupon DTO

Table 492 – Loyalty Coupon DTO

Type	Name	Description
string	Description	Coupon description.
datetime	ExpiryDate	Coupon expiry date.
KeyValuePairDTO[]	Attributes	List of objects representing additional properties.

Loyalty Transaction Summary DTO

Table 493 – Loyalty Transaction Summary DTO

Type	Name	Description
string	Description	Transaction description.
string	Location	Location of the transaction.
datetime	TransactionDate	Time of the transaction.
int	PointsUsed	The number of reward points used in the transaction.
int	PointsBalance	The remaining reward points balance

		of the loyalty account.
decimal	AmountSpent	The amount of currency used in the transaction.
decimal	AmountBalance	The remaining currency balance of the loyalty account.
KeyValuePairDTO[]	Attributes	List of objects representing additional properties.

Offer Product DTO

Table 494 – Offer Product DTO

Type	Name	Description
Bool	AddProductPermanently	If true then the product will remain in the basket, even if the offer no longer applies
List<**OfferProductDTO**>	Children	Any children (products). This is always structured as a group (e.g. Drinks) with children (e.g. Pepsi, Coke etc.)
string	Description	The description of the group/product
Decimal	Offerprice	The price that the product is offered at
Decimal	PreOfferPrice	The normal (pre-discount) price of the product
Int	ProductGroupId	The id of the group or product offer. This is the only value that gets passed back to the server if the product gets added to the basket.
Int	ProductOfferId	
OfferCondition	OfferConditionType	An enumeration used to represent an action performed as a result of a basket item message
OfferQualifyingRule	OfferQualifyingRuleType	
Int	ProductId	The id of the product (for

		reference only). This is the Id that gets added to the basket if the product is selected
String	CustomConfiguraitonAlias	
String	Title	The title of the group/product
String	ImageUrl	The image URL for offer product
String	LargeImageUrl	Large image URL for offer product

Provider Basket Item DTO

Table 495 – Provider Basket Item DTO

Type	Name	Description
Bool	AddProductPermanently	If true then the product will remain in the basket, even if the offer no longer applies
Decimal	BasePrice	
Int	BasketItemId	
String	ConfigurationState	
Int	ConfigurationStateTypeId	
Decimal	DiscountProductValue	
Bool	IsDeleteable	
Bool	IsDiscountable	
Bool	IsInserted	
Bool	IsMatched	
Bool	IsPercentageDiscount	
Bool	IsPotentialMatch	
Bool	IsSaleable	
Bool	IsUpdateable	
Bool	IsVisible	
OfferItemStatus	OfferAdded	

Int	OfferId	
Int	OfferProductGroupId	
OfferCondition	OfferConditionType	An enumeration used to represent an action performed as a result of a basket item message
OfferQualifying Rule	OfferQualifyingRuleType	An enumeration used to represent an action performed as a result of a basket item message
Int	OriginalBasketItemId	
Int	ParentId	
Decimal	PreOfferPrice	
Bool	PriceOverride	
Int	ProductConfigurationStateTypeId	
Int	ProductId	
String	ProductText	
Int	Quantity	
Bool	SetMatched	
Decimal	UnitPriceNet	

Invitee DTO

Table 496 – Invitee DTO

Type	Name	Description
Integer	InviteeId	The unique identifier of the invitee
String	FirstName	The firstname of the invitee
String	LastName	The lastname of the invitee
String	EmailAddress	EmailAddress of the invitee
String	Password	Password of the invitee(currently not implemented)
String	PasswordSalt	Not used
Integer	CustomerId	The Customer's (Event Coordinator) unique identifier who created this Invitee. If we are using single user model the invitee can be created before the customer login into the system.

Inviteegroup DTO

Table 497 – Inviteegroup DTO

Type	Name	Description
Integer	InviteeGroupId	The unique identifier of the inviteeGroup
Integer	CustomerId	The Customer's (Event Coordinator) unique identifier who created this InviteeGroup.
String	InviteeGroupName	Name of the InviteeGroup
String	InviteeGroupDescription	Description of the InviteeGroup
List<Invitee DTO>	InviteeCollection	List of invitees belongs to this group

Grouporder DTO

Table 498 – Grouporder DTO

Type	Name	Description
Integer	GroupOrderId	The unique identifier of the GroupOrder
Integer	BasketId	The unique identifier of the basket that is used to store all the items associated with this event.
Integer	CustomerId	The Customer's (Event Coordinator) unique identifier who created this GroupOrder.
String	GroupOrderName	Name of the event
DataTime	StartDateTime	Date OrderRequired
DateTime	CloseDate	Time when the event coordinator wants to close the event. The system will not place the order to the POS till the time reaches. But it will lock the user from making changes to their cart.
String	Note	Note added by event coordinator which will be added in the invitation email
Integer	LimitOrder	Order Limit for each invitee (It doesn't include tax)
String	Status	Status of the GroupOrder (Possible Values: Open, Scheduled, Cancelled, Failed, Completed, Closed).

String	StatusText	Status of the Order. Since the order will be hold in OHEICS database till the guests add their order, this column indicates the error description if it's not able to place an order.
String	BaseUrl	BaseUrl with which the GUId will be appended. Baseurl to the order page.
Integer	ClientOrderTypeId	Indicated the order type id (Order Class Enumeration)
GroupOrderInviteeDTO	GroupOrderInviteeCollection	List of invitees associated with the event
GroupOrderInviteeInstanceDTO	GroupOrderInviteeInstaceCollection	Instance of the invitees for a particular event
CustomerDTO	Customer	EventCoordiantor customer Object
Integer	PaymentMethodId	Mode of payment. Payment Method enumerator.
String	PaymentMethodText	Description of the payment mode.
Integer	ChannelPaymentMethodId	Unique Identifier to the ChannelPaymentMethod
String	ChannelPaymentMethodText	Description to the channelpaymentmethod
Integer	CustomerPaymentCardId	Unique identifier to the customerpayment card -If the payment mode is credit, card details will be stored in the customer paymentcard.
PaymentCardDTO	PaymentDTO	Stored CustomerPaymentCard object
Integer	ClientId	Unique Identifier to the client
Intger	StoreId	Unique Identifier to the store in which they are ordering from
Integer	ChannelId	Unique Identifier to the channel
Integer	BrandId	Unique Identifier to the brand
Integer	TradeZoneId	Unique Identifier to the TradeZone (Not Used: Currently, grouporder supports only collection.)
Guid	ApplicationInstanceID	Application Instance Id

Grouporder Invitee DTO

Table 499 – Grouporder Invitee DTO

Type	Name	Description
Integer	GroupOrderInviteeId	The unique identifier of the GroupOrderInvitee
Integer	GroupOrderId	The unique identifier of the GroupOrder
Integer	InviteeGroupId	The unique identifier of the InviteeGroup
Integer	InviteeId	The unique identifier of the Invitee
List<InviteeDTO>	InviteeCollection	List of invitees belongs to the grouporder
List<InviteeGroupDTO>	InviteeGroupCollection	List of inviteeGroup belongs to the grouporder

Grouporder Inviteeinstance DTO

Table 500 – Grouporder Inviteeinstance DTO

Type	Name	Description
Integer	GroupOrderId	The unique identifier of the GroupOrder
Integer	InviteeId	The unique identifier of the Invitee
InviteeDTO	Invitee	Invitee Object details
Integer	ItemCount	Number of items added to the basket by the invitee
Decimal	ItemTotal	Amount of the partial basket for the invitee
String	Status	Status of the invitee’s order (Possible Values: EmailSent,Created,Deleted,EmailFailed, ReminderSent,ReminderFailed, CheckOut)

Guid	InviteeGUID	Unique GUID that is associated with this instance
DateTime	OrderSentTime	EmailSent Time
String	Url	Url to the order page for this invitee

Client Culture DTO

Table 501 – Client Culture DTO

Type	Name	Description
String	CultureCode	Culture code, i.e. en-GB
String	CultureText	Display name of culture, i.e. English(UK)
Bool	IsDefault	Flag indicating whether this culture is default culture for client application.

Localization Configuration DTO

Table 502 – Localization Configuration DTO

Type	Name	Description
LocalisationAddressCategory	AddressCategory	The address category
OrderClass	OrderClass	The order class
List<LocalisationConfiguration ElementDTO>	Elements	Collection of localization configuration element

Localization Configuration Element DTO

Table 503 – Localization Configuration Element DTO

Type	Name	Description
LocalisationInputMode	InputMode	The element’s input mode
AddressSearchType	AddressField	The address field
AddressSearchType	ParentAddressField	Parent address field
AddressSearchType	TriggeredAddressField	The address field of the element, this element will trigger for the typeAhead data. This is derived from the parent relationship set

		against the type-ahead element.
Int	DisplayOrder	Order of display for this element
String	DisplayTitleText	Display title text for the element
String	DisplayLabelText	Display label text for the element
String	ValidationRegex	Validation regular expression
String	ValidationMessage	Regular expression validation message for the element
Bool	Enabled	Flag indicating whether element is enabled or not
Bool	IsMandatory	Flag indicating whether element is mandatory for input or not
String	MandatoryValidationMessage	Mandatory validation message if element input is mandatory
String	Value	Value of the element

Bucket Root Product Dto

Table 504 – Bucket Root Product Dto

Type	Name	Description
BucketProductConfigurationStateDto	ConfigurationState	The configuration state of the root bucket product in context.
Int	ProductId	The ProductId of the root product.
String	ProductName	The name of the root product.
Decimal	ProductPrice	The default price of the root product without additional selections.
Bool	IsValid	The validity of the configuration.
List<BucketProductStepDto>	Steps	A list of steps (selection containers) which are part of this particular root product configuration.
String	DisplayLabelText	Display label text for the element

String	ValidationRegex	Validation regular expression
String	ValidationMessage	Regular expression validation message for the element
Bool	Enabled	Flag indicating whether element is enabled or not
Bool	IsMandatory	Flag indicating whether element is mandatory for input or not
String	MandatoryValidation Message	Mandatory validation message if element input is mandatory
String	Value	Value of the element

Bucket Product Step DTO

Table 505 – Bucket Product Step DTO

Type	Name	Description
Int	ProductId	The ProductId of the step product.
List<BucketStepItem>	ChildItems	A list of product items contained in this step.
Int	RelationshipQuantity	This quantity is the number of selections required to make the configuration valid for this step.
String	Title	The title of the step product.
Int	TotalQuantitySelected	The total number of items selected in this step
Bool	InvalidQuantity	An indicator of the validity of the quantity currently selected.
String	Description	A product description for this step.
Bool	IsMandatory	An indication of whether this step is a mandatory step in the overall configuration
Int	ParentProductId	The product id of the root product.

Bucket Step Item

Table 506 – Bucket Step Item

Type	Name	Description
Int	ProductId	The ProductId of the step item product.
String	ItemTitle	The title of the step item.
String	MarketingDescription	A description for marketing and display purposes of the step item.
Decimal	DealPrice	The deal price of this step item.
String	Color	The color of this step item.
Bool	IsConfigurationRequired	An indicator stating whether configuration is required for this step item.
String	BackgroundImage	The location of the background image.
Int	MaxRequiredQuantity	The maximum number of items which may be selected of this type. A value of 1 typically indicates an exclusive selection.
Bool	IsDefaultItem	Indicates whether this step item is a default item in the step.
Int	ParentProductId	The product id of the parent step.
Bool	ItemSelected	Indicates whether this step item is selected.
Int	SelectedQuantity	The quantities of this item selected.

Bucket Configuration State DTO

Table 507 – Bucket Configuration State DTO

Type	Name	Description
Int	ConfigurationAliasId	The id of the configuration alias.
Bool	ContainerIsMandatory	Indicates whether this container is mandatory.
Int	ContainerQuantity	The quantity of the container state.

String	CustomConfigurationAlias	The name of the custom configuration alias
Int	InviteeId	Invitee Id.
String	InviteeName	Invitee name.
Bool	IsActiveConfigurationItem	Is the item active in this configuration.
Bool	IsConfigurable	Indicates whether configurable.
Bool	IsConfigured	Indicates whether configuration has taken place.
Bool	IsDefault	Indicates whether this is a default state item.
Guid	ProductConfigurationId	A global unique identifier for this product configuration.
String	ProductDescription	A description of this state product.
Int	ProductId	The product identifier for this state item.
Int	ParentId	The parent product identifier of this state item.
Int	Quantity	The quantity of this state item.
List<BucketProduct ConfigurationState Dto>	ChildItems	A list of descendant child items.

Order Options DTO

Table 508 – Order Options DTO

Type	Name	Description
Int	OrderOptionId	The id of the Order Option.
String	OptionDescription	The description of the Order Option.

CURRENCY DTO

The Currency DTO contains information about the currency associated with the Store (it may inherit from brand too) that allows a client application to determine the currency and show the currency specific information like currency symbol.

Table 509 – CURRENCY DTO

Type	Name	Description
string	CurrencyIsoCode	The Three-letter alphabetic codes that represent the currency following the standard specified by the International Organization for Standardization (ISO).
string	CurrencySymbol	A graphic symbol used as a shorthand for a currency's name, especially in reference to amounts of money.
string	CurrencyName	The name of the Currency.
string	BaseCulture	The default Culture Code linked with the currency.
string	CurrencyFormat	The format in which amount of money is shown.
bool	IsCurrencySymbolAfter	Indicates whether the CurrencySymbol need to be displayed after the amount of money.

Voucher Validation Details DTO

The Voucher Validation Details DTO contains information about valid Order Type, valid Channel Type, valid Date, and valid Stores. Example of usage of these fields: If users apply a voucher is not valid for the current selected store, voucher will be applied failed and this DTO will contain information of the valid store that users should apply voucher at.

Table 510 – VOUCHER VALIDATION DETAILS DTO

Type	Name	Description
List<OrderClass>	SupportedOrderTypes	Used to contains list of Order Class that are valid for the being applied voucher
List<SupportedChannel>	SupportedChannels	Used to contains list of Channels enum that are valid for the being applied voucher
ValidDateDTO	SupportedDates	Valid Date Times of the current being applied voucher

List<ValidStoreDTO>	SupportedStores	Used to contains list of Stores that are valid for the being applied voucher
List<VoucherValidationErrorDetailDTO>	VoucherValidationErrorDetails	Contains the DTO that provide information about the invalid business rules for voucher
List<SalesChannel>	SupportedSalesChannel	Sale Channel that are supported by vouchers

Valid Date DTO

Contains information about valid Day of Week, Start Date and End Date of voucher.

Table 511 – Valid Date DTO

Type	Name	Description
List<DayOftheWeek>	ValidDayOfWeek	Used to contains list of Order Class Day of Week enum that are valid for the being applied voucher
DateTime	StartDate	Valid Start Date of the current being applied voucher
DateTime	EndDate	Valid End Date of the currently being applied voucher

Valid Store DTO

Contain information about the valid store for the current being applied voucher. This DTO is used when a voucher is applied failed because it is not valid for the current selected store.

Table 512 – Valid Store DTO

Type	Name	Description
int	StoreId	Id of Store
string	StoreName	Display Text of Store

Voucher Validation Error Details DTO

Contains information about Error Code, Error Message when a voucher is failed to apply

Table 513 – Voucher Validation Error Details DTO

Type	Name	Description
VoucherErrorCode	ValidationErrorCode	Voucher Error Code enum
string	ValidationErrorMessage	Message provides details about the broken rules when applying voucher

Sales Channel

Contains information about sales channel that are supported by vouchers.

Table 514 – Sales Channel

Type	Name	Description
SupportedChannel	ChannelType	Supported channel type by vouchers
string	Channel	Channel Text

Qualifier Product DTO

Contains information about qualifier products that users' needs to add to their basket in order to qualify for the offer.

Table 515 – Qualifier Product DTO

Type	Name	Description
int	ProductId	ID of product
string	Title	Display text of product title
string	Description	Description of product
string	MarketingDescription	Marketing description of product
string	CustomConfigurationAlias	Custom configuration alias of product
decimal	Price	Product price
string	ImageUrl	URL of image of product

int	NeededQuantity	No of items should be added to basket in order to avail offers

Basket Voucher DTO

Contains information about qualifier products that users need to add to basket in order to avail offer.

Table 516 – Basket Voucher DTO

Type	Name	Description
Int	VoucherId	ID of the voucher
Decimal	VoucherDiscountAmount	Amount of discount
Bool	VoucherDiscountIsPercentage	Whether voucher is percentage or not
Bool	VoucherQualifies	True if voucher qualifies
Bool	IsVoucherApplied	True if voucher is applied
Bool	IsReApplyVoucher	Indicates whether Voucher of the old basket which has not check out is re-applied to the new basket
Bool	IsReApplyVoucherSuccess	Indicates whether voucher is re-applied successfully
Bool	IsVoucherValid	True if voucher code is valid
Bool	VouchersEnabled	True if voucher is enabled
Bool	VoucherActive	True if voucher is active
String	TermsAndConditionsText	Terms and conditions for the voucher
String	TermsAndConditionsUrl	Url that contains terms and conditions for the voucher
String[]	DisplayMessages	The validation result message
List<QualifierProductDTO>	VoucherQualifierProduct	List of qualifier product that users need to add into basket in order to avail offer
Decimal	VoucherQualifierAmount	The amount that users need to have in basket in order that avail offer
VoucherValidationDetailsDTO	VoucherValidationDetails	Contains information about conditions that users need in order

		to avail offer

Voucher DTO

The Voucher DTO contains information about the currency associated with the Store (it may inherit from brand too) that allows a client application to determine the currency and show the currency specific information like currency symbol.

Table 517 – Voucher DTO

Type	Name	Description
String	VoucherCode	Applied voucher code.
String	VoucherDescription	The description of voucher code.

Product Tag DTO

Table 518 – Product Tag DTO

Type	Name	Description
Int	ProductTagId	Product Tag
String	ImageUrl	Image URL for Product Tags
String	DisplayText	Display Text for Product Tags
String	Description	Description for Product Tags

Favorite DTO

The DTO contains information about basket or basket item collection that will be added to favorite.

Table 519 – Favorite DTO

Type	Name	Description
int	FavoriteId	Favorite Id
String	FavoriteName	Favorite name
Int	UserAccountId	User account id
Int	ClientId	Client id
String	ClientText	Client Text

Int	BrandId	Brand id
String	BrandText	Brand text
Bool	IsGroup	Is group
List<FavoriteItemDTO>	Items	Collection of Favorite Item

Favorite Item DTO

Table 520 – Favorite Item DTO

Type	Name	Description
Int	FavoriteItemId	Favorite item id
Int	FavoriteId	Favorite id
Int	ParentId	Parent favorite item id
Int	ProductId	Product id
Int	ParentProductId	Parent product id
String	ProductText	Product name
Int	Quantity	Quantity
String	Reference	Reference
String	ItemGroup	Item group
Int	ConfigurationStateTypeId	Configuration state type id
String	ConfigurationState	Configuration state
Int	ConfigurationId	Configuration id
List<FavoriteItemDTO>	Items	Collection of Favorite Item

5 Reference Types

General Enumerations

Order Class

An enumeration used to represent the type of order being placed.

Table 521 – Order Class Enumeration

Name	Int Value	Description
Collection	1	An order to be collected from the store
Delivery	2	An order to be delivered to the customers address

Customer Status Type

An enumeration used to represent the customer status.

Table 522 – Customer Status Type Enumeration

Name	Int Value	Description
NOT_SET	0	Default unset type
Normal	1	Normal
Greylisted	2	Greylisted. Customer can still place orders however all of them will require an approval from manager (callback required).
Blacklisted	3	Blacklisted. Orders cannot be taken for this customer.

Customer Note Type Class

An enumeration used to represent the customer note type classes.

Table 523 – Customer Note Type Class Enumeration

Name	Int Value	Description
NOT_SET	0	Default unset type
Discount	1	Discount
CustomerComplaint	2	Customer Complaint
Information	3	Information (general note)
OrderComplaint	4	Order Complaint

Callback Resolution

An enumeration used to represent callback order resolution.

Table 524 – Callback Resolution Enumeration

Name	Int Value	Description
Approve	0	Approve order
Reject	1	Reject order

Order Status Type

An enumeration used to represent order status.

Table 525 – Order Status Type Enumeration

Name	Int Value	Description
NOT_SET	0	Default unset type
OrderNew	1	New order
OrderCreated	2	Open order
CallBackRequired	3	Closed order
CallBackApproved	4	Canceled order
CallBackRejected	5	Voided order
POSPreCheckOnlineApproved	6	
POSPreCheckOnlineRejected	7	
POSPreCheckOrderValidApproved	8	
POSPreCheckOrderValidRejected	9	
PaymentApproved	10	
PaymentRequiresMoreDetail	11	
PaymentRejected	12	
LoyaltyPending	13	
LoyaltyApproved	14	
LoyaltyRejected	15	
POSOrderPlacementApproved	16	
POSOrderPlacementRejected	17	
OrderCancelled	18	

OrderSuccess	19
OrderVoid	20
PaymentPreAuthApproved	21
PaymentPreAuthRejected	22
PaymentfinalisationApproved	23
PaymentfinalisationRejected	24
POSPaymentCardRejected	31
POSUnknownError	32
POSCommunicationError	33
POSNetworkCommuicationError	34
POSConfigurationError	35
POSCheckInformationLineError	36
CateringOrderPlaced	37
CateringOrderWarningSent	38
CateringOrderConfirmed	39
CateringOrderCancelled	40
PaymentVoid	41

Order Status Class Type

An enumeration used to represent order status classes.

Table 526 – Order Status Class Type Enumeration

Name	Int Value	Description
NOT_SET	0	Default unset type
New	1	New order
Open	2	Open order
Closed	3	Closed order
Cancelled	4	Canceled order
Void	5	Voided order

Reason Code Type

An enumeration used to represent reason code types.

Table 527 – Reason Code Type Enumeration

Name	Int Value	Description
NOT_SET	0	Default unset type
CallOrderAbandonment	1	Call or Order Abandonment
Callback	2	Callback
CallbackRejectedReason	3	Callback reject reason
CallbackApprovedReason		Callback approved reason
CustomerNote		Customer general note
DiscountReason		Discount reason

Start Order Flow

An enumeration used to represent the time period the order is being placed for.

Table 528 – Start Order Flow Enumeration

Name	Int Value	Description
NOT_SET	0	Default unset type
ResolveOpenCalls	1	Resolve any existing open call.
SelectBrand	2	Select one of the brands available for the user.
SearchCustomer	3	Proceed to customer search and selection to start a new order

Fulfillment Time Type

An enumeration used to represent the time period the order is being placed for.

Table 529 – Fulfillment Time Type Enumeration

Name	Int Value	Description
ASAP	1	An order to be fulfilled as soon as possible
Advance	2	An order to be fulfilled at some point later the same day
Future	3	An order to be fulfilled at a date in the future. (Not the same day)

Title

An enumeration used to represent the title of a person.

Table 530 – Title Enumeration

Name	Int Value	Description
Mr	1	Mr title
Mrs	2	Mrs title
Miss	3	Miss title
Ms	4	Ms title
Dr	5	Doctor title
Prof	6	Professor title
Other	7	Other title

Payment Method Type

An enumeration used to represent the type of a payment method.

Table 531 – Payment Method Type Enumeration

Name	Int Value	Description
NOT_SET	0	Default unset method type
Cash	1	A cash payment
CreditDebitCard	2	A credit card or debit card payment
LoyaltyCard	3	A loyalty card payment
PayLater	4	Payment to be provided after the order has been placed.
MultiplePayment	5	Multiple types of payment, may be used for partial payment or group order.
BillToRoom	6	Bill towards room payments.

Payment Card Class

An enumeration used to represent the class of a payment card.

Table 532 – Payment Card Type Enumeration

Name	Int Value	Description
Credit	1	A credit card
Debit	2	A debit card

Loyalty	3	A loyalty or gift card

Payment Card Type

An enumeration used to represent the type of a payment card.

Table 533 – Payment Card Type Enumeration

Name	Int Value	Description
NOT_SET	0	Unset default value
Credit	1	Refers to credit cards.
Debit	2	Refers to debit cards.
Loyalty	3	Refers to loyalty cards.

Store Attribute Type

An enumeration used to represent the type of store attributes.

Table 534 – Store Attribute Type Enumeration

Name	Int Value	Description
NOT_SET	0	Unset default value
Services	1	Refer to services

Basket Item Message Type

An enumeration used to represent the type of basket item message.

Table 535 – Basket Item Message Type Enumeration

Name	Int Value	Description
NOT_SET	0	Unset default value
InvalidItem	1	An invalid item or item configuration
InvalidAmount	2	An invalid amount

Basket Item Message Action

An enumeration used to represent the type of basket item message action.

Table 536 – Basket Item Message Action Enumeration

Name	Int Value	Description
NOT_SET	0	Unset default value
ItemRemoved	1	Item removed from basket.

Surcharge Type

An enumeration used to represent the type of surcharge.

Table 537 – Surcharge Type Enumeration

Name	Int Value	Description
NOT_SET	0	Unset default value
Delivery	1	The delivery surcharge
Holiday	2	A holiday based surcharge
General	3	A general surcharge
Payment	4	A payment surcharge (debit/credit card charges etc.)

Address Type

An enumeration used to represent the type of an address.

Table 538 – Address Type Enumeration

Name	Int Value	Description
Residential	1	A residential address (customer)
Store	2	A store address location
Business	3	A business address (customer)
Delivery	4	A delivery address (customer)
Billing	5	A billing address (customer)

Basket Product Add Type

An enumeration used to represent the type of product to add to the current basket.

Table 539 – Basket Product Add Type Enumeration

Name	Int Value	Description
ProductId	1	The provided identifier will be used as a menu product Id.
WishlistId	2	The provided identifier will be used as a customer wishlist Id.
PreviousOrderId	3	The provided identifier will be used as a previous customer order Id.

Product Topping Class

Table 540 – Product Topping Class Enumeration

Name	Int Value	Description
Topping	1	Topping is of type Topping.
Sauce	2	Topping is of type Sauce.
Cheese	3	Topping is of type Cheese.

Product Configuration Actions

Table 541 – Product Configuration Actions Enumeration

Name	Int Value	Description
ComplexProductSelected	1	Initial product configuration action
SizeSelected	2	Product configuration action to select size of product.
BaseSelected	3	Product configuration action to select base of product.
CoverageSelected	4	Product configuration action to select coverage of product.
SpecialitySelected	5	Product configuration action to select specialty of product.
SectionSelected	6	Product configuration action to select section of product.
ToppingSelected	7	Product configuration action to select

		topping for product.
CancelConfiguration	8	Product configuration action to cancel configuration of product.
CompleteConfiguration	9	Product configuration action to complete configuration of product.

Address Search Type

Table 542 – Address Search Type Enumeration

Name	Int Value	Description
NOT_SET	0	Allow serialization if missing, and defines no address search type
None	1	No Address search type
District	2	District field
PostCodeOrZip	3	Postcode or zip field
Region	4	Region field
Sector	5	Sector field
Territory	6	Territory field
TownCity	7	Town city fields
StreetName	8	StreetName field
BuildingName	9	Building name field
Organizationname	10	Organization name field
BuildingLetter	11	Building letter field
BuildingNumber	12	Building number field
RoomNumber	13	Room number field
Intersection	14	Intersection field
Country	15	Country field
TaxNumber	16	Tax number field
NearbyLandmark	17	Nearby landmark name field
FloorNumber	18	Floor number field

Localization Address Category

Table 543 – Localization Address Category Enumeration

Name	Int Value	Description
NOT_SET	0	Allow serialization if missing, and defines no address search type
StoreLocatorAddress	1	Store locator address localization configuration
CustomerAddress	2	Customer address localization configuration
GuestAddress	3	Annonymous user address localization configuration
StoreAddress	4	Store Address localization configuration

Localization Input Mode

Table 544 – Localization Input Mode Enumeration

Name	Int Value	Description
NOT_SET	0	Allow serialization if missing, and defines no add ress search type
FreeText	1	Free text input
TypeAhead	2	Dynamic typeahead input

Product Configuration Validation

Table 545 – Product Configuration Validation Enumeration

Name	Int Value	Description
ValidationSuccess	0	Successful validation.
InvalidBaseTypeSelection	1	Base type is invalid
InvalidSizeSelection	2	Size is invalid.
InvalidCoverageSelection	3	Coverage is invalid.
InvalidSectionSelection	4	Section is invalid.
InvalidSpecialitySelection	5	Specialty is invalid.
InvalidToppingCombination	6	Topping selection is invalid.
InvalidNoSelectionChanges	7	Number of changes selection is invalid.

Supported Channel

Table 546 – Supported Channel Enumeration

Name	Int Value	Description
NOT_SET	0	No actual value is set.
CallCentre	1	Call Centre channel
WebSite	2	Website channel
TextMessage	3	Text message channel

Method Request Enumerations

Configuration Type

An enumeration used to get a specific type of configurations.

Table 547 – Configuration Type Enumeration

Name	Int Value	Description
StoreProviderKeyValue	0	Store provider Key-value attributes
StoreKeyValue	1	Store configuration key-value attributes

Get Customer Notes Options

Table 548 – Get Customer Notes Options Enumeration

Name	Int Value	Description
NOT_SET	0	No value set
ParentOnly	1	Return only parent customer notes
ChildOnly	2	Return only child customer notes (those with RootParent id not null)
ParentAndChild	3	Return all

Get Reason Code Options

Table 549 – Get Customer Notes Options Enumeration

Name	Int Value	Description
NOT_SET	0	No value set
GetByReasonCodeType	1	Return reason codes by reason code type
GetByCustomerNoteTypeId	2	Return reason codes by customer note type

Method Response Enumerations

Service Error Type

An enumeration used in all message response objects to represent the error status of the request.

Table 550 – Service Error Type Enumeration

Name	Int Value	Description
None	0	No error was encountered
ExceptionThrown	1	A system error occurred
SessionExpired	2	The customer session has expired. A new session will need to be created by calling the StartSession method of the Session Service.

Cancel Order Result Type

Table 551 – Cancel Order Result Type Enumeration

Name	Int value	Description
Success	0	Order cancelled successfully, no error was encountered.
TimeTooLate	1	Too late to cancel the order.
PosError	2	A POS error occurred during the cancelation process.
OrderAlreadyCancelled	3	The order was already cancelled

Login User Start Session Result Type

Table 552 – Login User Start Session Result Type Enumeration

Name	Int value	Description
Success	0	No error was encountered
InvalidUserNameOrPassword	1	Invalid user name and/or password
InvalidApplicationId	2	An invalid application id was provided in the request

StartSessionResultType

Table 553 – StartSessionResultType Enumeration

Name	Int Value	Description
Success	0	The method completed successfully
InvalidApplicationId	1	The application Id supplied was not valid

Store SearchResultType

Table 554 – Store SearchResultType Enumeration

Name	Int Value	Description
Success	0	The method completed successfully
InvalidSearchAddress	1	The customer address supplied was invalid. This may occur when the address is not specific enough for the order type, i.e. delivery orders require an address specific enough to identify a single address.

Store Start Order Result Type

Table 555 – Store Start Order Result Type Enumeration

Name	Int Value	Description
Success	0	The method completed successfully
InvalidMenu	1	
InvalidTime	2	The selected order date and time cannot be verified as appropriate for the selected store.
InvalidStore	3	The selected store is invalid.
InvalidSelection	4	One or more of the order start parameters (order type, time, store, and menu) are invalid.
StoreIsClosed	5	The selected store is currently closed.
StoreIsOffline	6	The selected store cannot be contacted (point of sale), and is therefore offline.
InvalidOrder	7	
UserActionRequired	8	Further action is required by the user. I.e. selecting a specific deal type or entering a callback number of order selections which cannot be processed via online ordering.

Basket Response Status

Table 556 – Basket Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.
BasketNotFound	1	The requested basket cannot be found.
InvalidQuantity	2	An invalid quantity has been provided in the request for the GetBasketAddProduct service method call.
InvalidProductId	3	An invalid product Id has been provided in the request for the GetBasketAddProduct service method call.
InvalidBasketItem	4	An invalid basket item has been provided in the request for the GetBasketRemoveItem service method call.

Place Order Status

Table 557 – Place Order Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.
FailBrokenBusinessRule	1	A business rule which validates all request data for the place order method has been broken. Details will be provided in the response via the validation messages.
FailException	2	A critical error occurred whilst attempting to place the order. Details will be provided in the response via the error messages.
CallBack	3	The order requires an operator to call back the user attempting to place the order.
NoValidSlotsFound	4	There are no valid collection/delivery slots available.
InvalidDeliveryAddress	5	The delivery address provided is invalid.

Register Response Status

Table 558 – Register Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.
UsernameAlreadyRegistered	1	The selected username has already been registered.

Get Customer Response Status

Table 559 – Get Customer Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.
CustomerNotLoggedIn	1	A customer must be logged in to recover customer data.

Update Customer Response Status

Table 560 – Update Customer Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.

Login Result Type

Table 561 – Login Result Type Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.
InvalidUsernameOrPassword	1	The user provided an invalid username or password.

Password Recovery Response Status

Table 562 – Password Recovery Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.

UsernameNotFound	1	The method could not find the provided username.

Forgot Password Response Status

Table 563 – Forgot Password Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.
UserNotFound	2	The method could not find the provided username.

Add Address Response Status

Table 564 – Add Address Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.

Update Address Response Status

Table 565 – Update Address Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.

Remove Address Response Status

Table 566 – Remove Address Response Status Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.

Get Store Order Options Result Type

Table 567 – Get Store Order Options Result Type Enumeration

Name	Int Value	Description
Success	0	The method completed successfully.
InvalidStoreIdentifier	1	The selected StoreId of the request is invalid.

Custom Order Filter Options

Table 568 – Custom Order Filter Options

Name	Int Value	Description
All	0	No filter, get all orders
RecentOrders	1	Recent orders only
WishList	2	Orders from wishlist

User Account Status Result Type

Table 569 – User Account Status Result Type Enumeration

Name	Int Value	Description
IsLocked	0	User’s account us locked.
IsActive	1	User’s account is active

Get Localization Configurations Response Status

Table 570 – Get Localization Configurations Response Status Enumeration

Name	Int Value	Description
Success	0	Requested configuration found
NoConfigurationFound	1	No configuration found

Hide Past Order Result Type

Table 571 – Hide Past Order Result Type Enumeration

Name	Int Value	Description
Success		Mark an order as hidden successfully
Failed		Mark an order as hidden failed

Voucher Discount Status Code

Table 572 – Voucher Discount Status Code Enumeration

Name	Int Value	Description
None	0	

OfferDoesNotQualifyWithGroup	1	Offer failed as the offer group is not qualified
OfferDoesNotQualifyWithAmount	2	Offer failed as the offer amount is not qualified