



WEB API

Technical Specification

FRM Platform

Date: June 19th, 2023

Version: 70



Document Information

Document Title:	SOAP API Technical Specification – FRM Platform
File Name:	FRM_API_SOAP_EN.docx
Author(s):	Avanade

Document Revision History

Version	Date	Changed By	Items Changed Since Previous Version
1	Apr 9, 2015	Avanade	Initial Version
2	Apr 13, 2015	Avanade	Data Model reference and methods updates
3	Apr 14, 2015	Avanade	Data Model reference and methods updates
4	Apr 20, 2015	Avanade	Data Model reference and methods updates
5	May 06, 2015	Avanade	Data Model reference and methods updates
6	May 20, 2015	Avanade	Data Model reference and methods updates
7	Jun 12, 2015	Avanade	3.1 Fan: added BirthCountryCode field 3.8.3 Enumerators: added Empty values 3.8.3.3 MemberType: added J1987 value

8	Sep 8, 2015	Avanade	3.8.3.3 MemberType: added Junior Member 3.8.3.6 MembershipType: added Junior Member, Junior Gifter
9	Sep 24, 2015	Avanade	3.2 Junior Fan 3.9.3.14 RelationshipType 4.3 Junior Fan Methods
10	Oct 9, 2015	Avanade	3.2 Junior Fan 3.9.3.14 RelationshipType
11	Feb 5, 2016	Avanade	3.10 Web Request
12	Apr 14, 2016	Avanade	3.9.3.13 RequestArea
13	Apr 14, 2016	Avanade	3.9.3.13 RequestArea
14	Jun 9, 2016	Avanade	3.6 Product 3.9.3 Enumerators 4.6 Product Method
15	Jun 30, 2016	Avanade	3.3 Membership 3.9.3.9 MeshSize
16	Jul 5, 2016	Avanade	4.6.4 updateProductStatus
17	Aug 2, 2016	Avanade	3.3. AutomaticRenewalActivated 3.6. Pin, Instant Issuing, FirstAcquisitionDate, TypographicNumber
18	Aug 17, 2016	Avanade	3.7. File 3.10.3.7. FileType enumerator
19	Sep 21, 2016	Avanade	Added privacy date and flag in the product object
20	Oct 11, 2016	Avanade	Add new values to MembershipStatus Enum
21	Oct 25, 2016	Avanade	Update fan model and add new enumerator FanType. Add new object Vro. Update the product data model. Add the method createGuestFan Add the method blockOldTdt Add the method updateFeedbackValue Add the method updateActivationLink Update the file methods Add a new value for FeedbackType enumerator
22	Oct 25, 2016	Avanade	Update fan data model for the guest fan Added field FanRegistered on fan data model
23	Nov 4, 2016	Avanade	Update the CardStatus and CardLogisticStatus enumerators

			Add the method transferTicket Add the method changeSubscriptionType Update the Product data model
24	Nov 28, 2016	Avanade	Add new field PackTrackingUrl on Order Removed the mandatory for the fields PackShipmentType, PackShipperName e PackTrackingCode
25	Dec 02, 2016	Avanade	Add new field Personalization on OrderLine Removed the mandatory for the fields ProductName2 e StartDate Update fan model and add new enumerators OrderType e OrderStatus Add the method updateOrder
26	Dec 05, 2016	Avanade	Add GoogleId field on Fan
27	Dec 12, 2016	Avanade	Modify a value for OrderStatus enumerator
28	Jan 12, 2017	Avanade	Added WebTV as new MembershipType
29	Jan 13, 2017	Avanade	Added WebTV as new MemberType
30	Jan 26, 2017	Avanade	Added Currency field to Membership, Order, OrderLine, Product and Ticket objects
31	Feb 1, 2017	Avanade	Changed the mandatory value for the FanType field of the Fan
32	Feb 6, 2017	Avanade	Added Cash on delivery fee and Shipment Phone to Order Changed the enumerator RequestArea Changed WebRequest model Changed the File model Added getWebRequestFiles method Changed createWebRequest method Removed uploadFile method Added uploadFanFile method Added uploadWebRequestFile method
33	Feb 22, 2017	Avanade	Changed Order Data Model
34	Feb 23, 2017	Avanade	Added getMembershipByFanId
35	Feb 28, 2017	Avanade	Added Currency values
36	Mar 13, 2017	Avanade	Added Product Amount field on Membership and Product
37	Mar 21, 2017	Avanade	Added Currency values
38	Mar 23, 2017	Avanade	Added Privacy1 and Privacy1Date to Membership

			Added Privacy1, Privacy1Date, Privacy2 and Privacy2Date to Ticket Changed the mandatory attribute for Ticket Added ExpirationDate to File Changed the mandatory attribute for File
39	Mar 24, 2017	Avanade	Added ReplacementTicketRequired on Product
40	May 19, 2017	Avanade	Added 3.11 Waiting List Added createWaitingList method
41	Jun 05, 2017	Avanade	Add a new value for PaymentMethod enumerator
42	Jun 20, 2017	Avanade	Added new value for MemberType enumerator Added new two values for MembershipType enumerator Added Personalization to Membership Data Model
43	Jul 05, 2017	Avanade	Added 3.12 Event Added new enumerator EventType Added 4.11 Event Methods
44	Jul 18, 2017	Avanade	Updated MemberType and MembershipType enumerator
45	Aug 11, 2017	Avanade	Added RightOfWithdrawalDate to Membership Data Model
46	Sep 22, 2017	Avanade	Added EventNameDetail and FanConfirmation to Event Data Model
47	Sep 26, 2017	Avanade	Replaced FanConfirmation with FanServiceSelection in the Event Data Model Added DateOfFanServiceSelection to Event Data Model
48	Nov 09, 2017	Avanade	Updated the mandatory attribute for Product Update Product Methods Update File Methods
49	Jan 24, 2018	Avanade	Updated CardStatus and CardLogisticStatus enumerators Update blockOldTdt method (change frmFanCardId input)
50	May 21, 2018	Avanade	Added Museum, Stadium, Store and MuseumLastVisitDate to Fan Data Model Added Museum, Stadium and Store to Event Data Model

			Added upsertFan and upsertGuestFan methods
51	Jun 04, 2018	Avanade	Added StadiumLastVisitDate and StoreLastVisitDate to Fan Data Model Added Other and OtherDescription to Event Data Model Updated MembershipStatus enumerator Update Membership Data Model changing the type of PackShipmentStatus from "string" to a new enumerator
52	Aug 03, 2018	Avanade	Added 3.13 Juventus TV Data Model Added 3.14 Lead Data Model Added new enumerators JuventusTVOperation, JuventusTVSubscriptionType and LanguageCode Added 4.12 Juventus TV Methods Added 4.13 Lead Methods
53	Sep 21, 2018	Avanade	Added createFanLight and updateFanLight Added getPrivacyFanList Updated the RequestArea enumerator
54	Sep 25, 2018	Avanade	Added the object PaginatedResponse Added the property PrivacyExpiration on the Fan Changed the method getPrivacyFanList
55	Oct 8, 2018	Avanade	Added the method updateSubscription
56	Nov 27, 2018	Avanade	Updated the property PrivacyExpiration values Added the property Privaci1 and Privacy1Date on Lead
57	May 8, 2019	Avanade	Added method upsertFanLight
58	August 7,2019	Avanade	Added Web Request Area
59	January 15,2020	Avanade	Added Account and JOFC Subscription Entities Added Account and JOFC Subscription Methods Added AccountType, ClubType, Status Enumerators
60	January 30, 2020	Avanade	Added details about Account and JOFC Subscription Data-Model
61	April 13, 2020	Avanade	Added rest methods according to the new post assessment architecture. Removed / revised some SOAP methods in function of the new architecture



62	May 12, 2020	Avanade	Added prod endpoint of new farm services
63	May 29, 2020	Avanade	Changed API endpoints FanID and Uuid on ORder DataModel mutually mandatory Edited Waiting list Edited Juventus TV Datamodel Edited Lead DataModel Addebd PT to languages Removed updateSubscription to Mapp Inserted OrderStatus and SubscriptionStatus enumerator for Juventus TV
64	September 2, 2020	Avanade	Added UUID field to Event <u>DataModel</u>
65	September 7, 2020	Avanade	Inserted upsertMembership, upsertJuventusCard on REST service upsertJCard on SOAP service
66	November 13, 2020	Avanade	Added ErrorCode table for UpsertJuventusTV, CreateWebRequest, UpsertLead and token API.
67	December 13, 2020	Avanade	Removed the mandatory of Buyer and Purchased Date, CreatedAt properties in upsertMembership and upsertJcard API, respectively.
68	June 8, 2021	Avanade	Added fields MembershipCode and IsHolder in UpdateEvent and CreateEvent
69	June 22, 2021	Avanade	Added object Shipment and field Status to UpsertMembership
70	November 21, 2021	Avanade	Added "coupon" attribute to upsertMembership. Added "subscriptionId" and "AutomaticRenewal" to upsertJuventusTv
71	August 18, 2022	Avanade	Added shipping fields to the Fan Data-Model

Table of Contents

1	Introduction.....	10
1.1	Circulation.....	10
1.2	Updates.....	10
2	API Usage.....	11
2.1	Prerequisites.....	11
2.2	Interaction with services.....	11
2.3	Best Practices.....	12
3	API Data-Model Reference.....	13
3.1	Fan.....	13
3.2	Order.....	16
3.3	OrderLine.....	18
3.4	Waiting List.....	19
3.5	Event.....	20
3.6	Lead.....	21
3.7	Account.....	21
3.8	Juventus TV.....	22
3.9	JOFC Subscription.....	23
3.10	Juventus Card.....	24
3.11	Miscellaneous.....	24
4	SOAP API Service Reference.....	36
4.1	Authentication Methods.....	36
4.2	Fan Methods.....	37
4.3	Order Methods.....	40
4.4	Event Methods.....	41
4.5	Account Methods.....	42
4.6	JOFC Subscription Methods.....	42
4.7	JuventusCard Methods.....	43
5	REST API Service Reference.....	43
5.1	Functionality of the service.....	43
5.2	Authentication Methods.....	44



5.3	Juventus.com Methods.....	47
5.4	AIC Methods	56
5.5	Membership Methods.....	58



1 INTRODUCTION

The FRM Platform offers an extensive API available via SOAP/REST over HTTP calls. The following document lists all available methods and provides useful hints on actual service usage.

1.1 Circulation

This document is proprietary and confidential. Distribution of this document is limited to active (current) Juventus suppliers and partners. Every copy of this document in all forms must be destroyed once the privileged relationship with Juventus ends (e.g. if the contract expires and the supplier ceases to be an active supplier).

1.2 Updates

Please verify you are reading the most up-to-date version of this document before committing to an integration implementation.



2 API USAGE

All SOAP services are available at:

<https://frm.juventus.com/soap/FrmService.svc>

<https://frm.juventus.com/mapp/MappService.svc>

All REST services are available at:

<https://frm.juventus.com:8081/api/> for AIC integration

<https://frm.juventus.com:8082/api/> for WebRequest, LeadService, JuventusTVService

For brevity the namespace for services and methods is omitted in the rest of the document.

2.1 Prerequisites

Upon API service activation for your company, you will receive, along with this document, two alphanumeric tokens and a self-signed certificate:

- the **apiKey**, which uniquely identifies your company account on the FRM Platform
- the **userKey**, which uniquely identifies an authorized user profile for your company on the FRM Platform
- the *.juventus.com certificate which must be installed in the trusted root certification authorities on the client machine

Services will execute in the authorization context defined by the account and user profile identified by the two tokens. Operations which are not allowed by this authorization context will generate an exception.

You will not be able to invoke any service successfully until you have the two tokens describe above and you have pre-authorized your server's public IP address.

Tokens and IP authorizations can be obtained via your usual points of contact with Juventus IT Department.

2.2 Interaction with services

Each operation requires an authorization token (see 4.1) that must be acquired at the beginning of the interaction session.

The token is **re-usable** among subsequent API method calls and, for security reasons, should be invalidated as soon as it is not used anymore. Idle tokens are automatically invalidated after 15 minutes.

2.3 Best Practices

Don't leave your session open unless you know there is some more work to do in a short time.

Expect and manage errors. Assuming an API call was completed only because a call was made is not just right. FRM might answer with an error – you must check it and take actions accordingly, not just blindly go ahead and most likely cause more problems.

Use an exception wrapper around your calls. Most modern development environments allow for exception handling, which greatly simplifies management of error conditions. Use it to your advantage, so that any unexpected condition will not result in your application losing control of what's happening.

Recommendations timeout settings. You'd have two timeouts set in your application one for how long it should take to connect to your remote host, and a timeout for how long the socket connection should wait for a response from the server.

We recommend to set a reasonable connection timeout value into your soap web services client at 30 seconds.

The recommended value to set for the default socket timeout is **5 minutes (300 seconds)**.

Use a timeout wrapper around your calls. If exception handling is used, a timeout will simply be yet another kind of trapped/managed exception (make sure it is). If exception handling cannot be used for any reason, you should at least consider a timeout protection against unending API calls. If your application "hangs" indefinitely waiting for a response, a periodic job could be spawned at find itself running at the same time of another copy of itself.

Never retry in a tight loop. Retrying is necessary, but retrying in a tight loop isn't the best practice. If a resource shortage or contention caused the error, retrying it immediately will most likely just return the same error, and it will also uselessly consume resources. Always wait for some time before.

Recommendations for retry and sleep. We recommend to use the following common-sense parameters to manage your retry logic.

1. Retry at least 6 times
2. Wait at least 30 seconds between retries
3. If after (at least) 6 retries with a 30 second interval among them, you still can't get the work done, reschedule the application 30 minutes later.

Unless you have specific reasons not doing so (e.g. a delivery that should never start after a certain time limit), use the loop above as many times as you see fit. Even a major disaster on Internet connectivity will last no more than a few hours (well, a major disaster maybe not, but in that case, there'll be something else to worry about). So if you are persistent enough, you'll get the job done sooner than you think.

3 API DATA-MODEL REFERENCE

3.1 Fan

Type	Fan		
	A fan		
Attributes	Type	Description	Optional
Address1At	xs:string	Type at for the primary address.	yes
Address1FullAddress	xs:string	Type the full primary address.	yes
Address1Number	xs:string	Type the number for the primary address.	yes
Address1City	xs:string	Type the city for the primary address.	yes
Address1Country	xs:string	Type the country for the primary address.	yes
Address1CountryCode	xs:string	Type the country code for the primary address	yes
Address1Region	xs:string	Type the region for the primary address.	yes
Address1Province	xs:string	Type the province of the primary address.	yes
Address1Street1	xs:string	Type the first line of the primary address.	yes
Address1PostalCode	xs:string	Type the ZIP Code or postal code for the primary address.	yes
Address3Telephone	xs:string	Type the phone number for the shipping	yes
Address3CountryCode	xs:string	Type the shipping country code	yes
Address3City	xs:string	Type the shipping city	yes
Address3Country	xs:string	Type the shipping country name	yes
Address3County	xs:string	Type the shipping county/region	yes
Address3Province	xs:string	Type the shipping region	yes
Address3Street1	xs:string	Type the first line of the shipping address	yes
Address3Street2	xs:string	Type the second line of the shipping address	yes
Address3PostalCode	xs:string	Type the ZIP Code or postal code for the shipping address.	
AppDownload	xs:boolean	Represent if the fan has downloaded mobile app	yes
BirthCity	xs:string	Type the birth city	yes
BirthCountry	xs:string	Type the birth country name	yes
BirthCountryCode	xs:string	Type the birth country code	yes

BirthProvince	xs:string	Type the birth province	yes
BirthRegion	xs:string	Type the birth region	yes
Birthday	xs:datetime	Enter the fan's birthday for use in customer gift programs or other communications.	yes
DigitalActivation	xs:boolean	Represent if the fan has a digital activation	yes
DocumentCountry	xs:string	Type the document country	yes
DocumentExpirationDate	xs:datetime	Type the document expiration date	yes
DocumentIssuedBy	xs:string	Type the document issued by	yes
DocumentNumber	xs:string	Type the document number	yes
DocumentOwner	xs:string	Type the document owner	yes
DocumentType	xs:string	Type the document type	yes
Email	xs:string	Type the primary email address for the fan.	no
EmailAddress2	xs:string	Type the secondary email address for the fan.	yes
EmailAddress3	xs:string	Type the membership email address for the fan.	yes
FacebookID	xs:string	Type the Facebook id	yes
FanCardNumber	xs:string	Type the fan card number	yes
FanID	xs:string	Unique identifier of the fan	yes (mandatory on update)
FirstName	xs:string	Type the fan's first name to make sure the fan is addressed correctly in sales calls, email, and marketing campaigns.	no
FiscalCode	xs:string	Type the fiscal code	yes
Gender	GenderType	Select the fan's gender to make sure the fan is addressed correctly in sales calls, email, and marketing campaigns.	yes
HomePhone	xs:string	Type a first phone number for this fan.	yes
HomePhone2	xs:string	Type a second home phone number for this fan.	yes
HomePhone3	xs:string	Type a third home phone number for this fan	yes
JobTitle	xs:string	Type the job title of the fan to make sure the fan is addressed correctly in sales calls, email, and marketing campaigns.	yes

LanguageCode	xs:string	Type the fan's language code	yes
LanguageName	xs:string	Type the fan's language name	yes
LastName	xs:string	Type the fan's last name to make sure the fan is addressed correctly in sales calls, email, and marketing campaigns.	no
MobilePhone	xs:string	Type the mobile phone number for the fan.	yes
MobilePhone2	xs:string	Type the second mobile phone number for the fan	yes
MobilePhone3	xs:string	Type the third mobile phone number for the fan	yes
Privacy1	xs:boolean	Represent whether the fan accepted the Service privacy agreement or not	yes
Privacy1Date	xs:datetime	Represent when the fan accepted or denied the Service privacy agreement	yes
Privacy2	xs:boolean	Represent whether the fan accepted the Profiling privacy agreement or not	yes
Privacy2Date	xs:datetime	Represent when the fan accepted or denied the Profiling privacy agreement	yes
Privacy3	xs:boolean	Represent whether the fan accepted the Marketing privacy agreement or not	yes
Privacy3Date	xs:datetime	Represent when the fan accepted or denied the Marketing privacy agreement	yes
TwitterID	xs:string	Type the Twitter id	yes
NewsletterRegistration	xs:boolean	Type the newsletter registration	yes
CouponReceived	xs:boolean	Type the coupon received	yes
FanType	FanType	Type the type of the fan	yes
GoogleId	xs:string	Type the Google + Id	yes
Museum	xs:boolean	Represent whether the fan has registered or has updated his/her data to the Museum	yes
Stadium	xs:boolean	Represent whether the fan has registered or has updated his/her data to the Stadium	yes
Store	xs:boolean	Represent whether the fan has registered or has updated his/her data to the Store	yes
MuseumLastVisitDate	xs:datetime	Represent when the fan has registered or has updated his/her data to the Museum	yes
StadiumLastVisitDate	xs:datetime	Represent when the fan has registered or has updated his/her data to the Stadium	yes
StoreLastVisitDate	xs:datetime	Represent when the fan has registered or has updated his/her data to the Store	yes
PrivacyExpiration	xs:string	Get the privacy expiration period: Today, 7 days, 3 months or 6 months	yes
UUID	xs:string	Unique identifier of the fan (in AIC)	no

3.2 Order

Type	Order		
	A purchase order		
Attributes	Type	Description	Optional
BillingAddress	xs:string	Type the full billing address	yes
BillingAddressNumber	xs:string	Type the number for the billing address	yes
BillingAt	xs:string	Type at for the billing address	yes
BillingCity	xs:string	Type the city for the billing address	yes
BillingPostalCode	xs:string	Type the postal code for the billing address	yes
BillingProvince	xs:string	Type the province for the billing address	yes
BillingRegion	xs:string	Type the state for the billing address	yes
BillingState	xs:string	Type the street for the billing address	yes
BillingStreet	xs:string	Type the full billing address	yes
CampaignCode	xs:string	Type the campaign's code	yes
Discount	xs:decimal	Type the discount	yes
DiscountCode	xs:string	Type the discount's code	yes
DiscountName	xs:string	Type the discount's name	yes
FanID	xs:string	Unique identifier of the fan	no* (mutually mandatory with Uuid)
Uuid	xs:string	Unique identifier of the fan (in AIC)	no* (mutually mandatory with FanId)
Number	xs:string	Unique identifier of the order	No
OrderDate	xs:datetime	Type the order date	no
OrderStatus	OrderStatus	Type the order's status	no
PackShipmentStatus	xs:string	Type the shipment status	no
PackShipmentType	xs:string	Identify the shipment type	yes

PackShipperName	xs:string	Identify the shipper name	yes
PackTrackingCode	xs:string	Identify the shipper tracking code	yes
PackTrackingUrl	xs:string	Identify the url to track the pack	yes
PayeeID	xs:string	Unique identifier of the payee fan	no
PaymentDate	xs:datetime	Identify the payment date	no
PaymentMethod	PaymentMethod	Type the payment method	no
PurchaseChannel	PurchaseChannel	Type the purchase channel	no
ReferenceCode	xs:string	Identify the payment reference code	yes
ShipmentAddress	xs:string	Type the full shipment address	yes
ShipmentAddressNumber	xs:string	Type the number for the shipment address	yes
ShipmentAt	xs:string	Type at for the shipment address	yes
ShipmentCity	xs:string	Type the city for the shipment address	yes
ShipmentPostalCode	xs:string	Type the postal code for the shipment address	yes
ShipmentProvince	xs:string	Type the province for the shipment address	yes
ShipmentRegion	xs:string	Type the state for the shipment address	yes
ShipmentState	xs:string	Type the street for the shipment address	yes
ShipmentStreet	xs:string	Type the full shipment address	yes
ShippingFee	xs:decimal	Type the shipping fee	no
StoreName	xs:string	Type the store name	no
TotalAmount	xs:decimal	Type the total amount	no
Type	OrderType	Unique type of the order	no
UserDevice	xs:string	Type the user's device	yes
Currency	xs:string	Type the currency. Accepted values are: AED ARS AUD BRL CHF EUR GBP JPY	yes

		MXN USD	
Cash on delivery fee	xs:decimal	Type the cash on delivery fee	yes
Shipment Phone	xs:string	Type the shipping phone	yes
ShippingName	xs:string	Type the shipping name	yes
ShippingSurname	xs:string	Type the shipping surname	yes
BillingCompanyName	xs:string	Type the billing company name	yes
BillingEmail	xs:string	Type the billing email	yes
BillingFiscalCode	xs:string	Type the billing fiscal code	yes
BillingVatNumber	xs:string	Type the Billing VAT Number	yes

3.3 OrderLine

Type	OrderLine		
	A purchase order line		
Attributes	Type	Description	Optional
Amount	xs:decimal	Type the order line amount	no
CategoryLevel1	xs:string	Type the name of the category level 1	no
CategoryLevel2	xs:string	Type the name of the category level 2	no
CategoryLevel3	xs:string	Type the name of the category level 3	no
Discount	xs:decimal	Type the discount	yes
DiscountCode	xs:string	Type the discount's code	yes
Number	xs:string	Unique identifier of the order	no
ProductCode	xs:string	Type the product code	no
ProductName1	xs:string	Type the primary name of the product	no
ProductName2	xs:string	Type the secondary name of the product	yes
ProductPrice	xs:decimal	Type the single product price	no
Quantity	xs:integer	Type the product quantity	no
StartDate	xs:datetime	Identify the available product date	yes
Supplier	xs:string	Type the supplier code	no
Personalization	xs:string	Type the description of product personalization	Yes

Currency	xs:string	Type the currency. Accepted values are: AED ARS AUD BRL CHF EUR GBP JPY MXN USD	yes
----------	-----------	---	-----

3.4 Waiting List

Not used at the moment

Type	WaitingList		
	A waiting list		
Attributes	Type	Description	Optional
FanID	xs:string	Unique identifier of the fan	no
Region	xs:string	Type the region of residence for this fan	no
Telephone	xs:string	Type the phone number for this fan	no
Quantity	xs:string	Type the tickets quantity	no
SubmissionDate	xs:datetime	Type the submission date	no
EventName	xs:string	Type the event's name	no
Sector1	xs:string	Type a first sector	yes*
Sector2	xs:string	Type a second sector	yes*
Sector3	xs:string	Type a third sector	yes*
FirstName1	xs:string	Type the first fan's first name	yes
LastName1	xs:string	Type the first fan's last name	yes
Birthday1	xs:datetime	Type the first fan's birthday	yes
BirthCity1	xs:string	Type the first fan's birth city	yes
Region1	xs:string	Type the first fan's region of residence	yes
FanCard1	xs:string	Type the first fan's Fan Card	yes
FirstName2	xs:string	Type the second fan's first name	yes
LastName2	xs:string	Type the second fan's last name	yes

BirthDay2	xs:datetime	Type the second fan's birthday	yes
BirthCity2	xs:string	Type the second fan's birth city	yes
Region2	xs:string	Type the second fan's region of residence	yes
FanCard2	xs:string	Type the second fan's Fan Card	yes
FirstName3	xs:string	Type the third fan's first name	yes
LastName3	xs:string	Type the third fan's last name	yes
BirthDay3	xs:datetime	Type the third fan's birthday	yes
BirthCity3	xs:string	Type the third fan's birth city	yes
Region3	xs:string	Type the third fan's region of residence	yes
FanCard3	xs:string	Type the third fan's Fan Card	yes
FirstName4	xs:string	Type the fourth fan's first name	yes
LastName4	xs:string	Type the fourth fan's last name	yes
BirthDay4	xs:datetime	Type the fourth fan's birthday	yes
BirthCity4	xs:string	Type the fourth fan's birth city	yes
Region4	xs:string	Type the fourth fan's region of residence	yes
FanCard4	xs:string	Type the fourth fan's Fan Card	yes
* At least one from Sector1, Sector2 or Sector3 is required			

3.5 Event

Type	Event		
	Event		
Attributes	Type	Description	Optional
ExternalID	xs:string	Unique identifier of external system	no
FanID	xs:string	Unique identifier of the fan	no
UUID	xs:string	Unique identifier of the fan	no (mandatory if FanID is not present)
EventType	EventType	Type the event's type	no

EventName	xs:string	Type the event's name	no
EventDate	xs:datetime	Type the event's date	no
ApplicationDate	xs:datetime	Type the application date	no
Attendance	xs:boolean	Represent whether the fan confirm to attend the event	yes
DateOfAttendanceConfirmation	xs:datetime	Represent when the confirm to attend the event	yes
EventNameDetail	xs:string	Type the event's name detail	yes
FanServiceSelection	xs:boolean	Represent whether the fan is selected to attend the event	yes
DateOfFanServiceSelection	xs:datetime	Represent when the fan is selected to attend the event	yes
Museum	xs:boolean	Represent whether the channel of fan data capture is Museum	yes
Stadium	xs:boolean	Represent whether the channel of fan data capture is Stadium	yes
Store	xs:boolean	Represent whether the channel of fan data capture is Store	yes
Other	xs:boolean	Represent whether the channel of fan data capture is Other	yes
OtherDescription	xs:string	Type the description of channel "Other"	yes
MembershipCode	xs:string	Type the membership Code associated to the event	yes
IsHolder	xs:string	Represents whether the fan is holder of the membership	yes

3.6 Lead

3.7 Account

Type	Account		
	An account		
Attributes	Type	Description	Optional
Account Name	xs:string	Type the name of the account.	no
Account Number	xs:string	Type the number of the account (unique identifier of the account)	no
Address1City	xs:string	Type the city for the primary address.	yes
Address1Country	xs:string	Type the country for the primary address.	yes
Address1country	xs:string	Type the ISO2 country code for the primary address.	no
Address1Region	xs:string	Type the region for the primary address.	yes
Address1Street1	xs:string	Type the first line of the primary address.	yes
Address1:ZIP/PostalCode	xs:string	Type the ZIP Code or postal code for the primary address.	yes

Address2City	xs:string	Type the city for the secondary address.	yes
Address2Country/Region	xs:string	Type the country for the secondary address.	yes
Address2country	xs:string	Type the ISO2 country code for the secondary address.	yes
Address2Street1	xs:string	Type the first line of the secondary address.	yes
Address2:ZIP/PostalCode	xs:string	Type the ZIP Code or postal code for the secondary address.	yes
AccountType	AccountType	Type the type of account (The allowed values are indicated in AccountType)	no
ClubPresident	xs:string	Type the FanID of the club president	yes
ClubSecretary	xs:string	Type the FanID of the club secretary	yes
ClubType	ClubType	Type the type of club (The allowed values are indicated in ClubType)	yes
EndDate	xs:datetime	Type the end date of the club	yes
Email	xs:string	Type the email of the the account	no
Fiscal Code	xs:string	Type the fiscal code of the account	yes
Phone	xs:string	Type the phone number of the account	yes
PEC	xs:string	Type the PEC of the account	yes
SDI Code	xs:string	Type the SDI code of the account	yes
Registered Name	xs:string	Type the business name of the account	no
VAT Number	xs:string	Type the VAT number of the account	Yes
Status	Status	Type the status of account (The allowed values are indicated in Status)	Yes

3.8 Juventus TV

Type	Juventus TV		
	A Juventus TV		
Attributes	Type	Description	Optional
UserID (Email/numerotelefono)	xs: string	This field is the userID and email or phone number may arrive	yes
Create Date	xs:datetime	Represents the creation date on MPP	no
Client UserId	xs: string	This is the FRM ID that will be used for reconciliation with the Fan in FRM.	yes
Last Updated	xs:datetime	Last update date on MPP systems	no

OrderID	xs: string	Unique identifier of the order	yes
Name (ava_name di Dynamics)	xs: string	Valued as "userid-orderid"	no
OrderStatus	OrderStatus	Represents the order status	no
SubscriptionStatus	SubscriptionStatus	Represents the subscription status	yes
Description	xs: string	Descriptive text field	no
OrderDate	xs:datetime	Date on which the order was placed	no
Prodotto	Prodotto		no
Subscription Created Date	xs:datetime	Subscription creation date	no
Subscription Last Update	xs:datetime	Subscription last update date	no
ServiceExpiry	xs:datetime	End of validity date	no
Currency	xs:string	Type the currency. The accepted values are: AED ARS AUD BRL CHF EUR GBP JPY MXN USD	no
Gross Amount	xs:decimal	The subscription amount	no

3.9 JOFC Subscription

Type	JOFC Subscription		
	A JOFC Subscription		
Attributes	Type	Description	Optional
Currency	xs:string	Type the currency. The accepted values are: AED ARS AUD BRL CHF EUR GBP JPY MXN USD	yes



FanID	xs:string	Type the FanID of the Fan (Unique identifier of the fan)	no
JOFC	xs:string	Type the Account Name (the unique identifier of the Account)	no
EndDate	xs:datetime	Type the subscription's end date	no
StartDate	xs:datetime	Type the subscription's start date	no
SubscriptionNumber	xs:string	Type the subscription Number (the unique identifier of the JOFC subscription)	no
TotalAmount	xs:decimal	Type the amount of the subscription in local currency	yes
TotalAmountBase	xs:decimal	Type the amount of the subscription in Euro	yes
Status	Status	Type the status of the subscription (The allowed values are indicated in Status)	yes

3.10 Juventus Card

Type	JOFC Subscription		
	A Juventus Card		
Attributes	Type	Description	Optional
Code	xs:string	Type the Unique Id of the Juventus Card	no
ActivationStatus	JcardStatus	Type the activation status of the JCard (The allowed values are indicated in JCardStatus)	no
PurchaseDate	xs:datetime	Type the purchase date	no
ExpirationDate	xs:datetime	Type the expiration date	yes
CreatedAt	xs:datetime	Type the creation date	yes
HolderFrmid	xs:string	Type the FanID of the Fan (Unique identifier of the fan)	yes (If it is present HolderUUID)
HolderUUID	xs:guid	Type the UUID of the Fan (Unique identifier of the fan)	Yes (If it is present HolderFrmID)
AccountNumber	xs:xs:string	Type the Account Number of the Account	no

3.11 Miscellaneous

3.11.1 AuthToken

Type	AuthToken		
	An authorization token		
Attributes	Type	Description	Optional
	xs:string	Token content	no

Content			
---------	--	--	--

3.11.2 Response

Type	Response		
	An operation response		
Attributes	Type	Description	Optional
Feedback	FeedbackType	Operation result	no
Values	xs:sequence	Operation return values	no
Message	xs:string	Operation message/error description	no

3.11.3 PaginatedResponse

Type	Response		
	An operation response with paginated records		
Attributes	Type	Description	Optional
Feedback	FeedbackType	Operation result	no
Values	xs:sequence	Operation return values	no
Message	xs:string	Operation message/error description	No
currentPage	xs:integer	The current page of record retrieved	No
pageSize	xs:integer	Number of records extracted per page	No
hasMoreRecords	xs:boolean	Indicates if there are more records to retrieve	no

3.11.4 Enumerators

3.11.4.1 FeedbackType

Type	FeedbackType	
	An operation result	
Description	Type	Value
	xs:integer	1

Empty		
Success	xs:integer	2
Failure	xs:integer	3
Not authorized	xs:integer	4

3.11.4.2 GenderType

Type	GenderType	
	Fan gender	
Description	Type	Value
Empty	xs:integer	0
Male	xs:integer	1
Female	xs:integer	2

3.11.4.3 FanType

Type	FanType	
	Fan type	
Description	Type	Value
Empty	xs:integer	0
Fan	xs:integer	100000000
JuniorFan	xs:integer	100000001
Prospect	xs:integer	100000002
Guest	xs:integer	100000003
Other	xs:integer	100000004

Commented [A1]:

3.11.4.4 JcardStatus

Type	JcardStatus	
	Activation Status	
Description	Type	Value
Empty	xs:integer	0
New	xs:integer	100000000

Pending_Documents	xs:integer	100000001
Pending_Documents_Validation	xs:integer	100000002
Pending_Delivery	xs:integer	100000003
Delivery_In_Progress	xs:integer	100000004
Delivered	xs:integer	100000005
Rejected	xs:integer	100000006
Active	xs:integer	100000007
Deactivated	xs:integer	5
Expired	xs:integer	6

3.11.4.5 FileType

Type	FileType	
	File Type	
Description	Type	Value
Empty	xs:integer	0
IdentityDocument	xs:integer	1
Photo	xs:integer	2

3.11.4.6 MeshSize

Type	MeshSize	
	MeshSize	
Description	Type	Value
Empty	xs:integer	0
S	xs:integer	100000000
M	xs:integer	100000001
L	xs:integer	100000002
XL	xs:integer	100000003

3.11.4.7 OrderType

Type	OrderType	
	Order type	
Description	Type	Value
Empty	xs:integer	0
Merchandising	xs:integer	100000000
Guest	xs:integer	100000001

3.11.4.8 OrderStatus

Type	OrderStatus	
	Order status	
Description	Type	Value
Empty	xs:integer	0
Defining	xs:integer	100000000
Waiting for payment	xs:integer	100000001
Successful payment	xs:integer	100000002
Order cancelled	xs:integer	100000003
Refunded	xs:integer	100000004
In refunding - cash on delivery	xs:integer	100000005

3.11.4.9 PaymentMethod

Type	PaymentMethod	
	Payment Method	
Description	Type	Value
Empty	xs:integer	0
Cheque	xs:integer	100000000
Fan-Card Activation	xs:integer	100000001
Postal	xs:integer	100000002
Bank Transfer	xs:integer	100000003
Gift Certificate	xs:integer	100000004

Credit Card	xs:integer	100000005
Cash	xs:integer	100000006
Mark	xs:integer	100000007
Scratch Card	xs:integer	100000008
VAT	xs:integer	100000009
PayPal	xs:integer	100000010
Cash On Delivery	xs:integer	100000011
Adyen	xs:integer	100000012

3.11.4.10 *ProductType*

Type	ProductType	
	Product type	
Description	Type	Value
Empty	xs:integer	0
Hurrà Juventus	xs:integer	100000000
Fan Card	xs:integer	100000001
Membership Voucher	xs:integer	100000002

3.11.4.11 *PurchaseChannel*

Type	PurchaseChannel	
	Purchase channel	
Description	Type	Value
Empty	xs:integer	0
Call Center	xs:integer	100000000
Internet	xs:integer	100000001
Event	xs:integer	100000002
Special Store	xs:integer	100000003

3.11.4.12 *TicketType*

Type	TicketType
------	------------

	Ticket type	
Description	Type	Value
Empty	xs:integer	0
Legend	xs:integer	100000000
Museum	xs:integer	100000001
Parking	xs:integer	100000002
Stadium	xs:integer	100000003

3.11.4.13 RequestArea

Type	RequestArea	
	Request Area	
Description	Type	Value
Match, Season Ticket & Parking	xs:integer	100000000
Juventus Membership & Club Doc	xs:integer	100000001
Museum & Stadium tour	xs:integer	100000002
Juventus Store & Products	xs:integer	100000003
Juventus Academy	xs:integer	100000004
Business Events	xs:integer	100000005
Others	xs:integer	100000006
Supporter Card Application Form	xs:integer	100000007
Supporter Card	xs:integer	100000008
Special Club	xs:integer	100000009
Season Ticket Request	xs:integer	100000010
Juventus Pass	xs:integer	100000011
Juventus TV	Xs:integer	100000012
Privacy	Xs:integer	100000013
Juventus Card	Xs:integer	100000014

3.11.4.14 RelationshipType

Type	RelationshipType	
	Relationship Type	
Description	Type	Value
Empty	xs:integer	0
Child	xs:integer	100000000
Nephew	xs:integer	100000001
Neighbor	xs:integer	100000002
Friend	xs:integer	100000003

3.11.4.15 EventType

Type	EventType	
	Event Type	
Description	Type	Value
Empty	xs:integer	0
Membership	xs:integer	100000000
Fan Engagement	xs:integer	100000001

3.11.4.16 PackShipmentStatus

Type	PackShipmentStatus	
	Pack Shipment Status	
Description	Type	Value
Empty	xs:integer	0
Sent	xs:integer	100000000
Awaiting Shipment	xs:integer	100000001
Waiting to be processed	xs:integer	100000002
Canceled	xs:integer	100000003

3.11.4.17 JuventusTVSubscriptionType

Type	EventType	
	JuventusTVSubscriptionType	
Description	Type	Value

Empty	xs:integer	0
Free Trial	xs:integer	100000000
Monthly	xs:integer	100000001
Annual	xs:integer	100000002
Special Offer	xs:integer	100000003

3.11.4.18 LanguageCode

Type	LanguageCode	
	LanguageCode	
Description	Type	Value
Empty	xs:integer	0
DE	xs:integer	100000000
EN	xs:integer	100000001
ES	xs:integer	100000002
FR	xs:integer	100000003
IT	xs:integer	100000004
ID	xs:integer	100000005
CN	xs:integer	100000006
JP	xs:integer	100000007
AR	xs:integer	100000008
PT	xs:integer	100000009

3.11.4.19 AccountType

Type	AccountType	
	the type of account	
Description	Type	Value
Empty	xs:integer	0
JOFC	xs:integer	100000000

3.11.4.20 ClubType

Type	ClubType	
	the type of account	
Description	Type	Value
Empty	xs:integer	0
Persona Fisica	xs:integer	100000000
No Profit	xs:integer	100000001

3.11.4.21 Status

Type	Status	
	the status of the account	
Description	Type	Value
Active	xs:integer	0
Inactive	xs:integer	1

3.11.4.22 Juventus TV SubscriptionStatus

Type	Status	
	Juventus TV Subscription Status	
Description	Type	Value
Active Subscription	xs:integer	0
Expired Subscription	xs:integer	1
Cancelled By AutoRenew Process	xs:integer	2
Cancelled By User	xs:integer	3
Cancelled By PayPal IPN	xs:integer	4
Failure Retry Mode	xs:integer	5
Cancelled By BACS	xs:integer	6
Subscription pending Start Date	xs:integer	7
Cancelled by SEPA	xs:integer	8

3.11.4.23 Juventus TV OrderStatus

Type	Status	
	Juventus TV Order Status	
Description	Type	Value
Cancelled	xs:integer	0
Completed	xs:integer	1
Refunded	xs:integer	2
Waiting	xs:integer	3
Max Refunded	xs:integer	4
Partial Refunded	xs:integer	5

3.11.5 Errors

Below the list of main custom error messages returned by operation results (see 3.8.2 *Response*), and their meaning.

Message	Meaning
Please provide apiKey and userKey	Credentials not specified
Invalid apiKey or userKey	Invalid credentials
Invalid authToken	Refresh the authorization token validity
Duplicate Fan	A fan with the specified criteria already exists. The existing fan is included in the operation return values.
Fan not found	Unable to retrieve fan with the specified criteria
Membership not found	Unable to retrieve membership with the specified criteria
Order not found	Unable to retrieve order with the specified criteria

4 SOAP API SERVICE REFERENCE

SOAP is a widely adopted standard protocol to enable, structured information exchange between heterogeneous systems via Web Services.

FRM web services are all provided with a single exposed web service descriptor (service name is svc:FRMService).

This document uses standard namespaces abbreviations as defined in:

- `{http://www.w3.org/2001/XMLSchema}` alias **xs**, used for standard XSD types

Location of the WS:

Service name	svc:FRMService
DEV	
Service Location	https://frm-dev.juventus.com/soap/frmservice.svc
WSDL Location	https://frm-dev.juventus.com/soap/frmservice.svc?wsdl
TEST	
Service Location	https://frm-test.juventus.com/soap/frmservice.svc
WSDL Location	https://frm-test.juventus.com/soap/frmservice.svc?wsdl
PROD	
Service Location	https://frm.juventus.com/soap/frmservice.svc
WSDL Location	https://frm.juventus.com/soap/frmservice.svc?wsdl

This chapter describes each of the FRM web services methods grouped in seven sections:

- Authentication Methods
- Fan Methods
- Membership Methods
- Order Methods
- Product Methods
- Ticket Methods
- Web Request Methods
- File Methods

4.1 Authentication Methods

Nearly all of FRM API operations require an authorization token. This is the way to programmatically authenticate/authorize your system to interact with the FRM API platform (see "2.1 Prerequisites").

The *Authentication Methods* allow token acquisition, refresh (extend its life) and invalidation.

4.1.1 borrowToken

Method	borrowToken	
	Returns a token to be used in the subsequent services invocations as a proof of service usage eligibility	
Input	Type	Description
apiKey	xs:string	Alphanumeric hash identifying a FRM customer
userKey	xs:string	Alphanumeric hash identifying a FRM user
Output	Type	Description
	Response	An authorization token response

4.1.2 keepaliveToken

Method	keepaliveToken	
	Refresh the authorization token validity to 15 minutes	
Input	Type	Description
token	AuthToken	The authorization token to be refreshed
Output	Type	Description
	xs:boolean	true if successful, false otherwise

4.1.3 invalidateToken

Method	invalidateToken	
	Revokes the validity of an authorization token	
Input	Type	Description
token	AuthToken	The authorization token to be revoked
Output	Type	Description
	xs:boolean	true if successful, false otherwise

4.2 Fan Methods

Fan Methods expose all Fan related operations:

- fan creation

- guest fan creation
- fan data retrieval
- fan data retrieval by fanCardNumber
- multiple fans data retrieval
- update fan information

4.2.1 createFan

Method	createFan	
	Creates a new Fan	
Input	Type	Description
token	AuthToken	An authorization token
fan	Fan	A fan
Output	Type	Description
	Response	The created fan response (with its proper identifier)

4.2.2 createGuestFan

Method	createGuestFan	
	Creates a new Fan of type Guest	
Input	Type	Description
token	AuthToken	An authorization token
fan	Fan	A fan
Output	Type	Description
	Response	The created fan response (with its proper identifier)

4.2.3 getFan

Method	getFan	
	Retrieves a specific Fan	
Input	Type	Description
token	AuthToken	An authorization token
fanIdentifier	xs:string	A fan identifier
Output	Type	Description

	Response	The retrieved fan response
--	----------	----------------------------

4.2.4 *getPrivacyFanList*

Method	getPrivacyFanList	
	Retrieves a list of Fans with privacy data expired	
Input	Type	Description
token	AuthToken	An authorization token
page	xs:integer	Number of the current records page to retrieve
Output	Type	Description
	PaginatedResponse	The retrieved fans response

4.2.5 *updateFan*

Method	updateFan	
	Updates a specific Fan	
Input	Type	Description
token	AuthToken	An authorization token
fan	Fan	A fan
Output	Type	Description
	Response	The updated fan response

4.2.6 *upsertGuestFan*

Method	upsertGuestFan	
	Create a guest fan if does not exist, otherwise update the existing Guest fan	
Input	Type	Description
token	AuthToken	An authorization token
fan	Fan	A fan
Output	Type	Description
	Response	The created/updated fan response

4.3 Order Methods

Order Methods expose all purchase order related operations:

- order creation
- order data retrieval
- order lines data retrieval
- update order information

4.3.1 createOrder

Method	createOrder	
	Creates a new Order	
Input	Type	Description
token	AuthToken	An authorization token
order	Order	A purchase order
orderLines	OrderLine	A list purchase order lines
Output	Type	Description
	Response	The created order response

4.3.2 getOrder

Method	getOrder	
	Retrieves a specific Order	
Input	Type	Description
token	AuthToken	An authorization token
orderNumber	xs:string	A purchase order identifier
Output	Type	Description
	Response	The retrieved order response

4.3.3 getOrderLines

Method	getOrderLines	
	Retrieves a specific Order Lines	
Input	Type	Description
token	AuthToken	An authorization token
orderNumber	xs:string	A purchase order identifier

Output	Type	Description
	Response	The retrieved order lines response

4.3.4 updateOrder

Method	updateOrder	
	Updates a specific Order	
Input	Type	Description
token	AuthToken	An authorization token
order	Order	A purchase order
Output	Type	Description
	Response	The updated order response

4.4 Event Methods

Event Methods expose all Event related operations:

- event creation
- update event information

4.4.1 createEvent

Method	createEvent	
	Creates a new Event	
Input	Type	Description
token	AuthToken	An authorization token
event	Event	An event
Output	Type	Description
	Response	The created event response

4.4.2 updateEvent

Method	updateEvent	
	Updates a specific Event	
Input	Type	Description
token	AuthToken	An authorization token
event	Event	An event

Output	Type	Description
	Response	The updated event response

4.5 Account Methods

Account Methods expose all Account related operations:

- account creation
- account update

4.5.1 createAccount

Method	createAccount	
	Creates a new Account	
Input	Type	Description
token	AuthToken	An authorization token
product	Account	An Account
Output	Type	Description
	Response	The created account response

4.5.2 updateAccount

Method	updateAccount	
	Update a specific Account	
Input	Type	Description
token	AuthToken	An authorization token
Account	Account	An Account
Output	Type	Description
	Response	The updated account response

4.6 JOFC Subscription Methods

JOFC Subscription Methods expose all JOFC Subscription related operations:

- JOFC Subscription creation
- JOFC Subscription update

4.6.1 createJOFCSubscription

Method	createJOFCSubscription
--------	------------------------



	Create a new JOFC Subscription	
Input	Type	Description
token	AuthToken	An authorization token
JOFC Subscription	JOFCSubscription	A JOFC Subscription
Output	Type	Description
	Response	The created JOFC Subscription response

4.6.2 updateJOFCSubscription

Method	updateJOFCSubscription	
	Update a specific JOFC Subscription	
Input	Type	Description
token	AuthToken	An authorization token
JOFC Subscription	JOFCSubscription	A JOFC Subscription
Output	Type	Description
	Response	The updated JOFC Subscription response

4.7 JuventusCard Methods

5 REST API SERVICE REFERENCE

Representational State Transfer (REST) is an architectural style for distributed systems. The REST architecture is based on HTTP. The operation provides a well-defined URL structure that uniquely identifies a resource or set of resources and the use of specific HTTP verbs for information retrieval (GET), for modification (POST, PUT, PATCH, DELETE) and for other purposes (OPTIONS, etc.).

5.1 Functionality of the service

The exposed service will be of the RESTFUL type, below we will analyze the call and answer details.

5.1.1 BASE URL



AIC Service

DEV Base URL <https://frm-dev.juventus.com:8081/api>
TEST Base URL <https://frm-test.juventus.com:8081/api>
PROD Base URL <https://frm.juventus.com:8081/api>

JCom Service

DEV Base URL <https://frm-dev.juventus.com:8082/api>
TEST Base URL <https://frm-test.juventus.com:8082/api>
PROD Base URL <https://frm.juventus.com:8081/api>

5.2 Authentication Methods

Nearly all FRM API operations require an authorization token. This is the way to programmatically authenticate/authorize your system to interact with the FRM API platform (see "2.1 Prerequisites").

The *Authentication Methods* allow token acquisition, refresh (extend its life) and invalidation.

5.2.1 borrowToken

Borrow Token Method returns a token to be used in the subsequent services invocations as a proof of service usage eligibility.

POST Request:

ENDPOINT: {{base_url}}/api/Token/BorrowToken

Input:

```
{
  "ApiKey": "1234b906ff4a13b79a4a837009333",
  "UserKey": "543214f92a4a53bf317069412345"
}
```

Output:

```
{
  "Feedback": 2,
  "Values": [
    {
      "Content":
        "123456HMcCtxn6CD5TTrR1sqou7Eun6iSgzQ+Vyvw8rNKhLteiYOWUclpRzye/Gsy50/vUbKq"
    }
  ]
}
```



```
wt+JryCxGCPT15SRcqtP+bYkLRUzmcVmUAcXdStiZWWSmAdrVJTGncNYsKy3/r9V+czLpwf
HZ2zvvhvxbOD+8bR4hlzjOPxRA="
  }
],
"Message": null,
"Error Code": 0

}
```

5.2.2 *keepaliveToken*

Keep Alive Token Method Refresh the authorization token validity to 15 minutes.

POST Request:

ENDPOINT: {{base_url}}/api/Token/KeepAliveToken

Input:

```
{
  "Content":
  "123456HMcCtxn6CD5TTrR1sqou7Eun6iSgzQ+Vyvw8rNKhLteiYOWUclpRzye/Gsy50//vUbKq
wt+JryCxGCPT15SRcqtP+bYkLRUzmcVmUAcXdStiZWWSmAdrVJTGncNYsKy3/r9V+czLpwf
HZ2zvvhvxbOD+8bR4hlzjOPxRA="
}
```

Output:

```
{
  "Feedback": 2,
  "Values": null,
  "Message": null,
  "ErrorCode": 0
}
```

5.2.3 invalidateToken

Invalidate Token Method Revokes the validity of an authorization token.

POST Request:

ENDPOINT: {{base_url}}/api/Token/InvalidateToken

Input:

```
{
  "Content":
  "123456HMcTxn6CD5TTrR1sqou7Eun6iSgzQ+Vyvw8rNKhLteiYOWUclpRzye/Gsy50//vUbqqt+JryCxGCPT15SRcqtP+bYkLRUzmcVmUAcXdStiZWWSmAdrVJTGncNYsKy3/r9V+czLpwfHZ2zvvhvxbOD+8bR4hlzjOPxRA="
}
```

Output:

```
{
  "Feedback": 2,
  "Values": null,
  "Message": null,
  "ErrorCode": null
}
```

5.2.4 Feedback Values

Key	Value
0	<i>Default</i>
1	<i>Empty</i>
2	<i>Success</i>
3	<i>Failure</i>
4	<i>Not authorized</i>
5	<i>Duplicate</i>



6	Timeout
---	---------

5.2.5 Error Messages and Error Codes

The table containing the mapping between the error code and the error message is shown below. In case no error is returned, the ErrorCode and the relative ErrorMessage are null.

Error Code	Message
400	Please provide ApiKey and UserKey
404	No token was passed to the method or the input argument was not a Token type
500	An Internal Error occurred.
501	Token is not in the correct format. Source is empty. Please check your user and api key
502	The token does not exist

5.3 Juventus.com Methods

5.3.1 Juventus TV

Juventus TV Methods expose all Juventus TV related operations:

- upsertJuventusTV

5.3.1.1 upsertJuventusTV

The upsertJuventusTV method will request the JSON object with:

- The Juventus TV object which will contain the information on the Juventus TV which will be upserted on FRM

The method will be invoked by following the following steps:

Invoke authentication services to generate a token. Once the token is obtained, you can call the method below, passing the token in the header.

POST Request:

ENDPOINT: `{{base_url}}/API/juventustv/UpsertJuventusTV`

Input

```
{
  "SubscriptionId": "Test1"
```

```

"UserID": "mario.rossi@email.com",
"CreateDate": "2020-03-25 12:55",
"HomeCountry": "Italy",
"BillingCountry": "Italy",
"UUID": "12345DDD-A52F-44C1-8078-E8BD8C432F8D",
"Name": "Test 1",
"OrderID": "Test 1",
"OrderStatus": 1,
"SubscriptionStatus": 2,
"Description": "Test description",
"OrderDate": "2020-03-23 13:55",
"Product": 3,
"SubscriptionCreateDate": "2020-03-24 13:55",
"SubscriptionLastUpdate": "2020-03-25 13:55",
"ServiceExpiry": "2021-03-25 13:55",
"Currency": 4,
"AutomaticRenewal": true,
"GrossAmount": 123.45
}

```

Nome colonna	Tipo	Required	Note
SubscriptionId	<i>string</i>	yes	
UserID	<i>string</i>	yes	
CreateDate	<i>string</i>	no	
HomeCountry	<i>string</i>	no	
BillingCountry	<i>string</i>	no	
UUID	<i>string</i>	yes	
Name	<i>string</i>	no	
OrderStatus	<i>int</i>	no	
SubscriptionStatus	<i>string</i>	yes	

Description	<i>string</i>	no	
OrderID	<i>string</i>	yes	
OrderDate	<i>string</i>	no	
Product	<i>int</i>	no	
SubscriptionCreatedDate	<i>string</i>	no	
SubscriptionLastUpdate	<i>string</i>	no	
ServiceExpiry	<i>string</i>	no	
Currency	<i>int</i>	no	
AutomaticRenewal	<i>bool</i>	no	
GrossAmount	<i>decimal</i>	no	

Output

```
{
  "Success": true,
  "ErrorMessage": null,
  "ErrorCode": null
}
```

Nome colonna	Tipo	Required	Note
Success	bool		
ErrorMessage	Varchar(250)		

The table containing the mapping between the error code and the error message is reported below. In case no error is returned, the ErrorCode and the relative ErrorMessage are null.

ErrorCode	Message
400	Token cannot be empty



401	Invalid token
501	A generic error occurred during UpsertJuventusTV: {exception message}
502	subscription field is mandatory
519	User ID is mandatory
520	UUID is mandatory
521	SubscriptionStatus is mandatory
522	OrderID ID is mandatory
523	An error occurred during UpsertJuventusTV: Fan with UUID {uuid} does not exist
526	Subscription ID is mandatory

5.3.1.2 Logic

The operation shows the following logic:

- Takes the Juventus TV passed in the request and checks that a FAN with digital identity (UUID) exists in FRM with the UUID transmitted.
- If the FAN is found with the same UUID in FRM, we proceed with verify that the transmitted orderID corresponds to a Juventus TV already present on FRM:
 - **Create:** if I don't find any Juventus TV I create it and associate it with the FAN
 - **Update:** if I find a Juventus TV I will update on it

5.3.2 Web Request

Web Request Methods expose all Web Request related operations:

- createWebRequest

5.3.2.1 createWebRequest

The createWebRequest method will request the JSON object with:

- Information on the UUID with which to associate the Web Request
- The Web Request object that will contain the Web request to be associated with the Fan and up to a maximum of 3 attachments with a maximum weight of 5mb each

The method will be invoked by following the following steps:

Invoke authentication services to generate a token. Once the token is obtained, you can call the method below, passing the token in the header.

POST Request:

ENDPOINT:

{{base_url}}/API/WebRequest/createWebRequest

Input

```
{
  "Area": 100000012,
  "datetime": "2020-03-25 12:55",
  "Description": "Test Invio Web request con 1 solo allegato",
  "Email": "prova1@test.it",
  "FirstName": "Giulio",
  "LastName": "Verdi",
  "uuid": "ADD74DDD-A52F-44C1-8078-E8BD8C432F8D",
  "Attachments":
    [{
      "FileName": "test allegato.txt",
      "FileSize": 16,
      "MimeType": "txt",
      "FanID": "FRM000000075319",
      "FileID": "123456_1",
      "FileType": 1,
      "ExpirationDate": "2020-04-04",
      "Content": "c3VjY2Vzcw=="
    },{
      "FileName": "test allegato.txt",
      "FileSize": 16,
      "MimeType": "txt",
      "FanID": "FRM000000075319",
      "FileID": "123456_1",
      "FileType": 1,
      "ExpirationDate": "2020-04-04",
      "Content": "c3VjY2Vzcw=="
    },{
      "FileName": "test allegato.txt",
      "FileSize": 16,
      "MimeType": "txt",
      "FanID": "FRM000000075319",
      "FileID": "123456_1",
```

```

    "FileType": 1,
    "ExpirationDate": "2020-04-04",
    "Content": "c3VjY2Vzcw=="
  }
}

```

Nome colonna	Tipo	Required	Note
uuid	string	yes	
datetime	string	yes	
hash	string	yes	
client_id	string	yes	
description	string	yes	
email	string	yes	
fanID	string	no	
firstName	string	yes	
lastName	string	yes	
area	string	yes	

Output

```

{
  "Success": true,
  "ErrorMessage": null,
  "ErrorCode": null
}

```

Nome colonna	Tipo	Required	Note
--------------	------	----------	------

Success	bool		
ErrorMessage	Varchar(250)		

The table containing the mapping between the error code and the error message is reported below. In case no error is returned, the ErrorCode and the relative ErrorMessage are null.

ErrorCode	Message
504	Email field is mandatory
510	WebRequest object is null. There are some serializations problem. Please check the object is compliant
511	Area field is mandatory
512	Description field is mandatory
513	FirstName field is mandatory
514	LastName field is mandatory
515	The file {FileName} exceeds 5 MB
516	The request contains more than 3 files
517	A generic error occurred during CreateWebRequest: {exception message}
525	An error occurred during CreateWebRequest: Fan with UUID: {Uuid} does not exist

5.3.2.2 Logic

The operation follows the following logic:

- Verify that the UUID camp is completed, otherwise KO returns with Success: **false**
- If the process is successful, an OK is returned, otherwise it will return a Success: **false**.

5.3.3 Lead

Lead Methods expose all Lead related operations:

- upsertLead

5.3.3.1 upsertLead

The upsertLead method will request the JSON object with:

- The Lead object which will contain the information on the Lead which will be upserted on FRM

The method will be invoked by following the following steps:

Invoke authentication services to generate a token. Once the token is obtained, you can call the method below, passing the token in the header.

POST Request:

ENDPOINT:

{{base_url}}/API/Lead/upsertLead

Input

```
{
  "Email": "giulio.corona@avanade.com",
  "Country": "Italy",
  "Language": "100000004",
  "Privacy3": "false",
  "Privacy3Date": "0001-01-01T00:00:00",
  "Privacy1": "true",
  "Privacy1Date": "0001-01-01T00:00:00"
}
```

Nome colonna	Tipo	Required	Note
Email	string	yes	
Country	string	no	
Language	string	no	
Privacy3	string	yes	
Privacy3Date	string	yes	
Privacy1	string	yes	
Privacy1Date	string	yes	

Output

```
{
  "Success": true,
  "ErrorMessage": null,
  "ErrorCode": null
}
```

Nome colonna	Tipo	Required	Note
Success	bool		
ErrorMessage	Varchar(250)		

The table with the mapping between the error code and the error message is shown below. In case no error is returned, the ErrorCode and the relative ErrorMessage are null.

ErrorCode	Message
400	Token cannot be empty
401	Invalid token
503	lead field is mandatory
504	Email field is mandatory
506	Privacy1 field is mandatory
507	Privacy3 field is mandatory
508	Privacy1Date field is mandatory
509	Privacy3Date field is mandatory
518	A generic error occurred during UpsertLead: {exception message}
524	An error occurred during UpsertLead: Lead with email: {email} is already a fan

5.3.3.2 Logic

The operation shows the following logic:

- Takes the lead passed in the request and checks that there are no fans with digital identity (UUID) already present for that email
- If finds a fan with digital identity with that email he answers ko to the caller without processing the record
- Otherwise:
 - Takes the Lead within the Request and verify (via email) that a Lead with that same email does not exist within FRM
 - If it exists, update the Lead, otherwise create it.
 - If a Fan without digital identity is found in FRM with the same Lead email the Lead is Qualified. Moreover, the fan acquired the Privacy flags (Service and Marketing) and the Privacy Marketing and Service Date of the Lead (in case of higher values)

5.4 AIC Methods

AIC Methods expose all AIC related operations:

- upsertFan

5.4.1 upsertFan

5.4.1.1 Method

The method will be invoked by following the following steps:

POST Request:

ENDPOINT:

{{base_url}}/webhook/user/upsertFan

Input

```
{
  "uuid":"12345678-9098765-432123456"
  "operation": UPDATE,
  "attributes": [
    "attr1",
    "attr2"
  ]
}
```

Nome colonna	Tipo	Required	Note
UUID	string	yes	

Operation	<i>string</i>	yes	
Attributes	<i>String[]</i>	no	Defines which fan attributes have been changed

Output

```
{
  "message": "Ok",
  "code": 1,
  "error": false,
}
```

Nome colonna	Tipo	Required	Note
Message	string		
Code	Int		
Error	string		

5.4.1.2 Logic

The operation shows the following logic:

- Takes the WebHook passed in the request and check that a Fan with a digital identity and status other than deleted that has the same UUID exists in the System.
 - If no fans are found with the communicated UUID, create the fan and return OK
 - If a Fan with UUID is found and status other than deleted, update the Fan and return OK
 - If a Fan with UUID is found and was equal to eliminated, KO returns

5.5 Membership Methods

5.5.1 *upsertMembership*

The method will request the JSON object which contains data about:

- Membership data
- Holder (with or without digital identity),
- Buyer
- Manager
- Product, Order and Order Item related to the Membership.

The method will be invoked by the following steps:

POST Request:

ENDPOINT: `{{base_url}}/api/upsertMembership`

Input

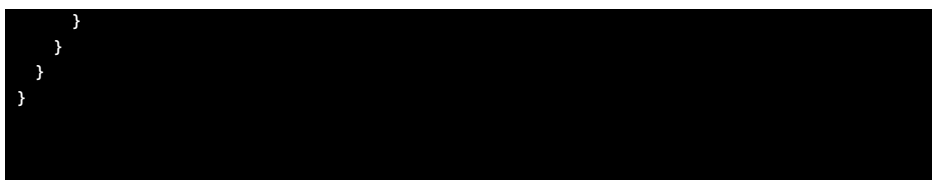
```
{
  "code": "JM21-XXXXXX",
  "status": "active",
  "validFrom": "2020-07-06T13:00:14.674Z",
  "validTo": "2023-07-06T13:00:14.674Z",
  "gift": false,
  "automaticRenewal": true,
  "createdAt": "2023-07-06T13:00:14.674Z",
  "holderFirstname": "Gabriele",
  "holderLastname": "Manna",
  "holderDob": "2008-02-19T00:00:00.000Z",
  "holderBirthCountry": "ITALIA",
  "holderBirthProvince": "ANCONA",
  "holderBirthCity": "ANCONA",
  "holderBirthCountryCode": "IT",
  "holderBirthProvinceCode": "AN",
  "holderBirthCityCode": "A271",
  "holderGender": "M",
  "holderEmail": "mangu73@gmail.com",
  "holderPhone": "393316819977",
  "buyer": {
    "providerId": "09e11ebd-bf61-408e-805c-95367910ea0e"
  },
  "manager": {
    "providerId": "09e11ebd-bf61-408e-805c-95367910ea0e"
  },
  "holder": {
    "providerId": "09e11ebd-bf61-408e-805c-95367910ea0e",
    "firstName": "marty",
    "lastName": "mcfly",
    "gender": "M",
    "taxCode": "MCFMTY90H30L219H",
    "birthCity": "TORINO",
    "birthCityCode": "L219",

```

```

    "birthProvince": "TORINO",
    "birthProvinceCode": "TO",
    "birthCountry": "ITALY",
    "birthCountryCode": "IT",
    "dob": "1990-06-30T00:00:00.000Z",
    "createdAt": "2020-07-06T13:00:14.674Z",
    "email": "marty.mcfly@dispostable.com",
    "phone": "00393487798464"
  },
  "product": {
    "sku": "j1897_2021_main",
    "name": "J1897",
    "season": "2021",
    "price": "120",
    "currency": "EUR",
    "description": "J1897 non per regalo",
    "validFrom": "2020-07-06T13:00:14.674Z",
    "validTo": "2023-07-06T13:00:14.674Z",
    "availableFrom": "2020-07-06T13:00:14.674Z",
    "availableTo": "2023-07-06T13:00:14.674Z"
  },
  "orderItem": {
    "id": 123,
    "qty": 1,
    "price": 50,
    "coupon": "A23VE-127AV"
    "shipment": {
      "status": " order_ready_to_ship"
    },
    "currency": "EUR",
    "createdAt": "2020-07-06T13:00:14.674Z",
    "rowTotal": 50,
    "voucher": {
      "code": "12345"
    },
  },
  "ShippingAddress": {
    "Street": "via delle vie",
    "PostalCode": "10100",
    "City": "TORINO",
    "State": "TO",
    "CountryCode": "IT"
  },
  "order": {
    "incrementId": "JMO20201000204",
    "paymentType": "paypal",
    "createdAt": "2020-07-06T13:00:14.674Z",
    "grandTotal": 50,
    "currency": "EUR",
    "itemsQty": 2,
    "voucher": {
      "code": "12345"
    }
  }
}

```



Nome colonna	Tipo	Required	Note
code	<i>string</i>	Y	
status	<i>string</i>	Y	
validfrom	<i>Datetime</i>	Y	
validto	<i>Datetime</i>	Y	
gift	<i>Boolean</i>		
automaticRenewal	<i>Boolean</i>		
createdAt	<i>Datetime</i>		
holderFirstname	<i>string</i>	*	Must be filled if the holder has not digital identity
holderLastname	<i>string</i>	*	Must be filled if the holder has not digital identity
holderDob	<i>string</i>	*	Must be filled if the holder has not digital identity
holderBirthCountry	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthProvince	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthCity	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthCountryCode	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthProvinceCode	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthCityCode	<i>string</i>		Field eventually filled if the holder has not digital identity
holderGender	<i>string</i>		Field eventually filled if the holder has not digital identity
holderEmail	<i>string</i>		Field eventually filled if the holder has not digital identity
holderPhone	<i>string</i>		Field eventually filled if the holder has not digital identity
buyer	<i>object</i>		
providerId	<i>string</i>		UUID

manager	object	Y	
providerId	string	Y	UUID
holder	object	Y	
providerId	string	*	UUID
firstName	string		
lastName	string		
gender	string		
taxCode	string		
birthCity	string		
birthCityCode	string		
birthProvince	string		
birthProvinceCode	string		
birthCountry	string		
birthCountryCode	string		
dob	Datetime		
createdAt	Datetime		
email	string		
phone	string		
product	object	Y	
sku	string	Y	
name	string	Y	
season	string	Y	
price	string	Y	
currency	string	Y	
description	string		
validFrom	Datetime	Y	
validTo	Datetime	Y	
availableFrom	Datetime		
availableTo	Datetime		
orderItem	object	Y	
id	int	Y	
qty	Int	Y	
price	Double	Y	
coupon	string		

shipment	object		
status	string		
currency	string		
createdAt	Datetime	Y	
rowTotal	Int	Y	
voucher	object		
code	string		
shippingAddress	object		
street	string		
postalCode	string		
city	string		
state	string		
countryCode	string		
order	object	Y	
incrementid	int	Y	
paymentType	string	Y	
createdAt	Datetime	Y	
grandTotal	Double	Y	
itemsqty	int	Y	
voucher	object		
code	string		

Output

```
{
  "Success": true,
  "ErrorMessage": null
}
```

Nome colonna	Tipo
Success	bool
ErrorMessage	Varchar(250)



5.5.2 Logic

This operation follows the following logic:

- Verify that Holder, Handler are existing on FRM (via UUID) , otherwise KO returns with Success=**false**
- If the Holder providerid is not present in the Request and firstname, lastname and dob are correctly populated a new fan will be created on FRM. A new FrmID will be generated.
- If the process is successful, an OK is returned, otherwise it will return the property Success=**false**.

The related process on D365 will verify:

- *If an order with the same number exists on FRM the order will be updated*
- *If an orderline with the same number exists on FRM the orderline will be updated*
- *If a Membership with the same number exists on FRM the Membership will be updated*
- *If a Membership Product with the same number exists on FRM the Membership Product will be updated*



5.5.3 upsertJuventusCard

The method will request the JSON object which contains data about:

- Juventus Card
- Holder (with or without digital identity)
- Customer
- Order and Order Item related to the JCard.

The method will be invoked by following the following steps:

POST Request:

ENDPOINT: {{base_url}}/api/upsertJuventusCard

Input

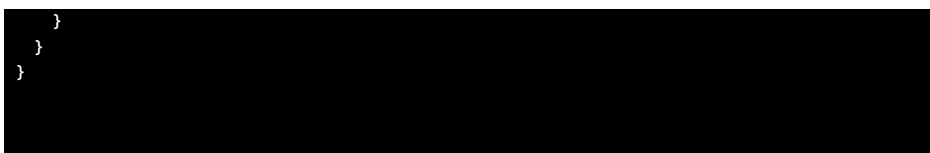
```
{
  "id": 49,
  "serialNumber": "014312774706",
  "status": "DELIVERED",
  "rejectionCause": "invalid_id_document",
  "deactivationReason": null,
  "purchaseDate": "2020-07-20T08:03:57.961Z",
  "expirationDate": "2030-07-19T22:00:00.000Z",
  "holderFirstname": "Cornelio",
  "holderLastname": "Silla",
  "holderDob": "1962-10-24T00:00:00.000Z",
  "holderBirthCountry": "ITALIA",
  "holderBirthProvince": "ROMA",
  "holderBirthCity": "ROMA",
  "holderBirthCountryCode": "IT",
  "holderBirthProvinceCode": "RM",
  "holderBirthCityCode": "A370",
  "holderGender": "M",
  "holderEmail": "poiuy@mailier2.net",
  "holderPhone": null,
  "holderTaxCode": "SLLCNL62R24H501M",
  "createdAt": "2020-07-20T08:04:00.742Z",
  "updatedAt": "2020-07-22T12:48:31.167Z",
  "deletedAt": null,
  "holder": {
    "id": 218,
    "customerId": "7542529",
    "providerId": "73f97881-d12f-4a98-833d-f5068b5e4770",
    "frmId": null,
    "firstName": "Cornelio",
    "lastName": "Silla",
    "gender": "M",
    "taxCode": "SLLCNL62R24H501M",
    "birthCity": "ROMA",
    "birthCityCode": "A370",
    "birthProvince": "ROMA",
    "birthProvinceCode": "RM",
```



```

    "birthCountry": "ITALIA",
    "birthCountryCode": "IT",
    "dob": "1962-10-24T00:00:00.000Z",
    "email": "poiuy@mailier2.net",
    "phone": null
  },
  "customer": {
    "id": 218,
    "customerId": "7542529",
    "providerId": "73f97881-d12f-4a98-833d-f5068b5e4770",
    "frmId": null,
    "firstName": "Cornelio",
    "lastName": "Silla",
    "gender": "M",
    "taxCode": "SLLCNL62R24H501M",
    "birthCity": "ROMA",
    "birthCityCode": "A370",
    "birthProvince": "ROMA",
    "birthProvinceCode": "RM",
    "birthCountry": "ITALIA",
    "birthCountryCode": "IT",
    "dob": "1962-10-24T00:00:00.000Z",
    "email": "poiuy@mailier2.net",
    "phone": null
  },
  "orderItem": {
    "id": 123,
    "qty": 1,
    "price": 50,
    "currency": "EUR",
    "createdAt": "2020-07-06T13:00:14.674Z",
    "rowTotal": 50,
    "voucher": {
      "code": "12345"
    },
  },
  "ShippingAddress": {
    "Street": "Via Italia 15",
    "PostalCode": "00046",
    "City": "Roma",
    "State": "RM",
    "CountryCode": "IT"
  },
  "order": {
    "incrementId": "JMO20201000204",
    "paymentType": "paypal",
    "createdAt": "2020-07-06T13:00:14.674Z",
    "grandTotal": 50,
    "currency": "EUR",
    "itemsQty": 2,
    "voucher": {
      "code": "12345"
    }
  }
}

```



Nome colonna	Tipo	Required	Note
id	<i>string</i>	Y	
serialNumber	<i>string</i>	Y	
status	<i>Datetime</i>	Y	
rejectionCause	<i>Datetime</i>	Y	
deactivationReason	<i>Boolean</i>		
purchaseDate	<i>Datetime</i>		
expirationDate	<i>Datetime</i>		
holderFirstname	<i>string</i>	*	Must be filled if the holder has not digital identity
holderLastname	<i>string</i>	*	Must be filled if the holder has not digital identity
holderDob	<i>Datetime</i>	*	Must be filled if the holder has not digital identity
holderBirthCountry	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthProvince	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthCity	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthCountryCode	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthProvinceCode	<i>string</i>		Field eventually filled if the holder has not digital identity
holderBirthCityCode	<i>string</i>		Field eventually filled if the holder has not digital identity
holderGender	<i>string</i>		Field eventually filled if the holder has not digital identity
holderEmail	<i>string</i>		Field eventually filled if the holder has not digital identity
holderPhone	<i>string</i>		Field eventually filled if the holder has not digital identity
holderTaxCode	<i>string</i>		
createdAt	<i>Datetime</i>		
updatedAt	<i>Datetime</i>		

deletedAt	<i>Datetime</i>		
holder	<i>object</i>	Y	
id	<i>string</i>		
customerId	<i>string</i>		
providerId	<i>string</i>	*	UUID
frmlId	<i>string</i>		
firstName	<i>string</i>	*	
lastName	<i>string</i>	*	
gender	<i>string</i>		
taxCode	<i>string</i>		
birthCity	<i>string</i>		
birthCityCode	<i>string</i>		
birthProvince	<i>string</i>		
birthProvinceCode	<i>string</i>		
birthCountry	<i>string</i>		
birthCountryCode	<i>string</i>		
dob	<i>Datetime</i>	*	
email	<i>string</i>		
phone	<i>string</i>		
customer	<i>object</i>	Y	
id	<i>string</i>		
customerId	<i>string</i>		
providerId	<i>string</i>	Y	UUID
frmlId	<i>string</i>		
firstName	<i>string</i>		
lastName	<i>string</i>		
gender	<i>string</i>		
taxCode	<i>string</i>		
birthCity	<i>string</i>		
birthCityCode	<i>string</i>		
birthProvince	<i>string</i>		
birthProvinceCode	<i>string</i>		
birthCountry	<i>string</i>		
birthCountryCode	<i>string</i>		

dob	<i>Datetime</i>		
email	<i>string</i>		
phone	<i>string</i>		
orderItem	<i>object</i>	Y	
id	<i>int</i>	Y	
qty	<i>Int</i>	Y	
price	<i>Double</i>	Y	
currency	<i>string</i>		
createdAt	<i>Datetime</i>	Y	
rowTotal	<i>Int</i>	Y	
Shipment	<i>object</i>		
Status	<i>string</i>		
ShippingAddress	<i>object</i>		
Street	<i>string</i>		
PostalCode	<i>string</i>		
City	<i>string</i>		
State	<i>string</i>		
CountryCode	<i>string</i>		
voucher	<i>object</i>		
code	<i>string</i>		
order	<i>object</i>	Y	
incrementid	<i>int</i>	Y	
paymentType	<i>string</i>	Y	
createdAt	<i>Datetime</i>	Y	
grandTotal	<i>Double</i>	Y	
itemsqty	<i>int</i>	Y	
voucher	<i>object</i>		
code	<i>string</i>		

Output

```
{
  "Success": true,
  "ErrorMessage": null
}
```

Nome colonna	Tipo
Success	bool
ErrorMessage	Varchar(250)

5.5.4 Logic

This operation follows the following logic:

- Verify that mandatory fields are always filled, otherwise KO returns with Success: **false**
- Verify that Holder and Customer are existing on FRM (via UUID), otherwise KO returns with Success: **false**
- If the Holder providerid does not exist in FRM and firstname, lastname and dob are correctly populated in the Request a new fan with FRMID will be created on FRM
- If the process is successful, an OK is returned, otherwise it will return a Success: **false**.

The process on D365 will verify:

- If an order with the same number exists on FRM the order will be updated
- If an orderline with the same number exists on FRM the orderline will be updated
- If a Juventus Card with the same number exists on FRM the Juventus Card will be updated

5.6 Event Methods

5.6.1 UpsertEvent

The method will request the JSON object which contains data about:

- Event type, name and details
- Customer
- Flag which represent if the event is to create or to update

POST Request

ENDPOINT: {{base_url}}/api/Event/UpsertEvent

Input

```
{
  "ExternalID": "",
  "FanID": "",
  "UUID": "",
  "EventType": "",
  "EventName": "",
  "EventDate": "",
  "ApplicationDate": "",
  "Attendance": ,
}
```

```

    "DateOfAttendanceConfirmation": "",
    "EventNameDetail": "",
    "FanServiceSelection": ,
    "DateOfFanServiceSelection": "",
    "Museum": ,
    "Stadium": ,
    "Store": ,
    "Other": ,
    "OtherDescription": "",
    "MembershipCode": "",
    "IsHolder": ,
    "ToCreate":
}

```

Nome colonna	Tipo	Required	Note
ExternalID	<i>string</i>	Y	
FanID	<i>string</i>	Y	If missing UUID is required
UUID	<i>string</i>	Y	If missing FanID is required
EventType	<i>EventType</i>	Y	
EventName	<i>string</i>	Y	
EventDate	<i>DateTime</i>	Y	
ApplicationDate	<i>DateTime</i>	Y	
Attendance	<i>Boolean</i>		
DateOfAttendanceConfirmation	<i>DateTime</i>		
EventNameDetail	<i>string</i>		
FanServiceSelection	<i>Boolean</i>		
DateOfFanServiceSelection	<i>DateTime</i>		
Museum	<i>Boolean</i>		
Stadium	<i>Boolean</i>		
Other	<i>Boolean</i>		
OtherDescription	<i>string</i>		
MembershipCode	<i>string</i>		
IsHolder	<i>Boolean</i>		
ToCreate	<i>Boolean</i>	Y	True = Create, False = Update

Output

```
{
  "Success": true,
  "ErrorMessage": null,
  "ErrorCode": null
}
```

5.6.2 New Error Codes

- ExternalIdMandatory = 527,
- EventTypeMandatory = 528,
- EventNameMandatory = 529,
- EventDateMandatory = 530,
- ApplicationDataMandatory = 531,
- ToCreateMandatory = 532

5.6.3 New Event Type

- Stadio : 100000002

5.6.4 Logic

This operation follows the following logic:

- Verify that mandatory fields are always filled, otherwise KO returns with Success: **false**
- Verify that Customer is existing on FRM (via UUID or FanID), otherwise KO returns with Success: **false**
- Verify that ToCreate flag is filled, otherwise KO return with Success: **false**
- If the process is successful, an OK is returned, otherwise it will return a Success: **false**.

The process on D365:

- If ToCreate flag is true the event will be created, otherwise, if ToCreate flag is false, the event will be updated