SERVER API CONCEPT

Smartico system provides two integration points - server to server API that is working over HTTP and JavaScript SDK.

- Server-to-server API is used to report update of the user profile and action of the users in a secure way
- JavaScript SDK is used to identify online status of the user, show "Popups" and provide Gamificaton interface

SERVER-TO-SERVER INTEGRATION

All server-to-server events should be reported via POST HTTP call with Basic Authorization.
A JSON structured payload of event should be placed in the post body.
Authorization token is specific for your Label and should be kept in the secure way.

Example of HTTP request

```
curl --request POST \
--url 'https://apis.smartico.ai/api/external/events/v2' \
--header 'Authorization: 32b18199-229b-4081-8106-864d4f759ed6' \
--header 'Content-Type: application/json'
```

End-point to send events: **https://apis.smartico.ai/api/external/events/v2**
Your current label: **DMO/Multi-Masters**
Your authorization token: **32b18199-229b-4081-8106-864d4f759ed6**

Important
As there is no way to revert accidently reported events, do not report events from QA/Staging environment to the production label. Request your account manager to create a separate label for QA purpose.

GENERAL STRUCTURE OF EVENT

Each event reported to API should be structured in the following way.

Field	Data type	Description
eid	string	Unique event ID generated by the integrated system
event_date	timestamp/integer	Event timestamp. UTC time with milliseconds
ext_brand_id	string	Mandatory for multi-brand setup, unique representation of brand in the integrated system. Each brand should be preliminary registered in Smartico system with specified external brand id. You have following brands registered under DMO/Multi-Masters label. **ZImmer11** - ext_brand_id: not defined **Cooktech** - ext_brand_id: not defined **SlotsDelight** - ext_brand_id: 1769 **MotorSich** - ext_brand_id: not defined **SmarticoWeb** - ext_brand_id: not defined **SpaceX** - ext_brand_id: spacex
user_ext_id	string	Unique id of user in the integrated system. Should be unique in the context of integration or in case of multi-brand setup - unique for the specific brand. Smartico will always trim and lower case. Be sure to use the same **user_ext_id** in the API and in the front-end JavaScript integration.
event_type	string	The event type, one of the registed on Smartico side. See the list of allowed events below. The name is case sensetive.

Example


```
{
    "eid": "ce94b181-868d-4291-b676-ed0fc5be77a0",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {}
}
```


To support bulk submission of events, the API is designed to accept an array of events or just one event.

Important
You can run up to 12 parallel requests to Smartico API with total limit of 6000 requests per minute, please be sure you are not exceeding this limit.
To avoid requests rejections please submit events in the batches and await for the the response. The maximum size of one HTTP post request is 10 MB, which is arround 4000 events with average size of 2.5KB


```
[
    {
        "eid": "08317fab-9302-4e3d-b8f0-8b110949a262",
        "event_date": 1757602862847,
        "ext_brand_id": "your_brand_ext_id",
        "user_ext_id": "some_user_id_1",
        "event_type": "update_profile",
        "payload": {}
    },
    {
        "eid": "024703c8-2780-4134-8275-6289a5c6f229",
        "event_date": 1757602862847,
        "ext_brand_id": "your_brand_ext_id",
        "user_ext_id": "some_user_id_1",
        "event_type": "update_profile",
```

```
        "payload": {}
    }
]
```

In the response Smartico will report general status of request and if there are failed events, list of such events with error for each one

```
{
    "err_code": 0,
    "failed_events": {
        "18f9d50c-7f6f-4ba8-9b4d-563073d3031e": [
            "Wrong format of event_date",
            "Missing required field deposit_amount in the event payload"
        ]
    }
}
```

RETRY MECHANICS

Integrated system should implement retry logic to deliver events through API.
Its recommended to use exponential increase of retry interval for next attempts starting from 1 second for the first retry.
If retry interval is reached 32 seconds, to conitnue with fixed 32 seconds interval.

DEDUPLICATION OF EVENTS

The data source should provide deduplication of the events. This means that all "transactional" data (bets, deposits, purchases etc) should be represented only once and be immutable.
All user profile updates reported with "update_profile" event should be provided only if there is at least one change in the property.
All events should have an "event_date" that represents the actual date of the business event.
In case of the "batch" reporting, the API calls should be distributed over the time.

Not following this approach may lead to inconsistent performance of the system and wrong execution of the campaigns, missions, tournaments, and automation rules.

BATCH UPDATES

In case of the "batch" reporting of historical transactions or update in the user profile, the API calls should be distributed over the time.
For example, you have nightly process that is recalculating VIP levels of users and updating user profiles.
When reporting such updates to Smartico, you should distribute API calls over the time, in order to avoid delays of execution of other Marketing & Gamification activities.

OPERATIONS ON PROPERTIES

By default any event like update_profile is updating listed properties of the user profile with new values.
In addition, there are specific operations that can be applied to the properties, like clearing the value, adding/removing value from the array etc.

Setting 3 tags in the core_external_markers property, overrides previous values

```
{
    "eid": "e2262b8c-af66-4477-97c3-9471f9cb6345",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "core_external_markers": [
            "tag1",
            "tag2",
            "tag3"
        ]
```

```
    }
}
```

Removing all values from the core_external_markers property

```
{
    "eid": "5a1b8fae-daf8-4c37-8369-52a435e02b23",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "!core_external_markers": null
    }
}
```

Adding tagX and tagY to the core_external_markers property

```
{
    "eid": "dffae827-feec-4558-954a-517f575fbe36",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "+core_external_markers": [
            "tagX",
            "tagY"
        ]
    }
}
```

Removing tag2 and tag3 from the core_external_markers property

```
{
    "eid": "bba9da6b-923d-4d30-ac6c-19159b9480e9",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "-core_external_markers": [
            "tag2",
            "tag3"
        ]
    }
}
```

Removing all tags that are starting with 'apple' or ending with 'juice'

```
{
    "eid": "8e927745-c04c-4e76-bb2e-7f645bff21ff",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "^core_external_markers": [
            "apple*",
            "*juice"
        ]
```

```
    }
}
```

You can also combine operations, e.g. comand below is removing all tags starting with 'apple' and after that adding 2 new tags

```
{
    "eid": "70f4cab0-05e0-406f-9002-4e3e7ad8e6bb",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "^core_external_markers": [
            "apple*"
        ],
        "+core_external_markers": [
            "tag1",
            "tag2"
        ]
    }
}
```

Important
Keep in mind that:
1. Setting a **null** to the property is not clearing the value. You should use '!property_name' to clear it.
2. You can use '+', '-', '^' operations only for properties that are defined as **array** of strings or array of numbers.
3. You can use '!' (clear operation) on any type of the property

SKIPPING EVENTS PROCESSING

In some cases you will want to populate user state on Smartico side, but not to trigger logic of marketing campaigns automation rules, missions, tournaments etc.
This could be relevant for the case when you want to populate history of pasts activity.
In order to skip CJM logic, you should add **skip_cjm** flag to the event payload.

```
{
    "eid": "70121e31-654f-4a15-9da6-e4617fa714a1",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_deposit_approved",
    "payload": {
        "skip_cjm": true
    }
}
```

FAQ AND GENERAL ADVICES

- Be sure you first create a user profile with 'update_profile' and only after that you are reporting any other activities
  Otherwise, you will get a rejection that the user doesn't exist.
- If you need to submit historical data, please do it in batches of ±1000 events per request, this way you will be able to submit historical data much faster.
- Be careful with submitting transactional data (deposits, bets, etc.). Smartico system doesn't have a deduplication mechanism, so if you submit the same event twice, it will be processed and stored twice.
  This also means that you should submit only the final states of transactions, e.g. if in your system the Deposit transaction has the state 'Pending' and then 'Approved', and it's not changing after 'Approved', you should submit only the 'Approved' state to Smartico.
  For the user profile data submitted with the 'update_profile' event, Smartico will keep only the last value of each property.

- Some user profile properties have strict validations for the possible values, for example, user_country, core_user_language, core_wallet_currency and core_account_status. You will get error if you will try to submit wrong value for such properties.
- If you want to clear the user profile property to the default value, you should submit null value for it.
- You can see how the events are arriving in the system and how the properties are stored. Go to the CRM tab, find the needed user, and check the 'User state' and 'Events' sections.
  You can also use Marketing \ Segments to build different segments of users based on the properties and events. For example, you can check the number of users with registration country 'DE' to validate that it's matching to your expectations.
- Note that properties of the user profile will become visible in the Smartico BackOffice for Segmentation and in the 'User state' in 24 hours after you set them for at least one user.
  If you want to see them immediately, please contact your Account Manager.
- Smartico has 2 flows of creating a user profile.
  1. We create a user profile when getting via API 'update_profile' event.
  2. We are also creating 'empty' user profile when the user is logged in on your website and you have Smartico JavaScript SDK installed.
  With the 2nd flow, the profile is 'empty', because it has only information that can be taken from the browser session, like the domain of the website, the language of the user, and unique ID (user_ext_id).
  At this stage Smartico still doesn't have information like registration date, country, currency etc, that should be submitted through the API with 'update_profile' event.
  From this perspective you should consider following:
  1. Try to submit 'update_profile' for newly registered users as soon as possible, so their profiles created from the JavaScript SDK will stay 'empty' for a short time.
  2. Set 'core_registration_date' property with the first 'update_profile' event. Smartico is using this property to track the quality of integration, by looking that there are no 'empty' users staying in the system.
- Monetary values should be reported with decimail point, e.g. as 12.35 for 12$ and 35c instead of 1235.

USER PROFILE CONCEPT

Smartico is keeping user profile updated by events sent to the API. The main event to create/update user profile is **update_profile**. Through this event Integrated system can update **any** property of the user profile defined on Smartico side.

Example of event structure

```
{
    "eid": "053b68ef-025f-4842-8c31-124e267df0fe",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "user_country": "BG",
        "core_user_gender": "F"
    }
}
```

In case user doesn't exist on Smartico side, it will be created automatically.

All properties of update_profile event are optional, this means that you can pass only those properties which were updated instead of passing full state of user every time.

IDENTIFYING USER

user_ext_id field of event is used to uniquelly identify user of integrated system.
This id should be unique in the context of specific brand, but not mandatory to be unique across the label.
This means that under one label you could have brands A and B, and both could have users John1985.

CURRENCY AND MONETARY VALUES

Smartico is designed to handle users with different currencies under one label - some users could have thier monetary activities tracked in USD, others in EUR etc.
The currency of user should be set before sending any monetary events or updating any monetary properties.

The currency of user cannot be changed once it defined.
Currency of the user can be also set together with monetary action.
All monetary values should be submitted in the whole units, e.g. 3.5$ should be reported as 3.5 and not as 350.

Deposit event that is also defining user currency

```
{
    "eid": "ae55828c-fac8-48de-ab64-5488ceef23a6",
    "event_date": 1757602862847,
    "ext_brand_id": "your_brand_ext_id",
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_deposit_approved",
    "payload": {
        "acc_last_deposit_amount": 100,
        "acc_last_deposit_date": 1655462961795,
        "core_wallet_currency": "USD"
    }
}
```

Important
API will reject events that are changing user currency when it was already defined.
While API will accept update of monetry property without prior setting user currency, this is highly not recommneded, as the marketers won't be able to treat properly such activities.

DOMAIN SPECIFIC EVENTS

Integrated system can also send other events specific to the business domain of the product, usually these are events like **Deposit**, **Casino Bet**, **Sport Bet**, **Open Position** etc.

Such events have mandatory fields that are essential for them, for example Deposit event should bring amount and the date/time of deposit.
Still these events can update any other properties of user profile.
For example "Deposit" event can also update a "User real money balance" property of user profile.

JACKPOTS API

In order to run Jackpots on Smartico platform, you should implement Casino Bets submission to the Jackpots API.
The end-point of the Jackpots API is different from the main API, while the authorization token is the same.
Please get familiar with Jackpots functionality before starting the integration

- Jackpots documentation - operational guide about setting up Jackpots
- Games catalog API - required to limit jackpot only to specific games
- Bonus API - required to issue winnings
- Jackpots on Expo - examples of the front-end integration
- Public API - reference for the front-end API

Jackpots API is providing 2 methods:
- Jackpots list - to get the list of available Jackpots
- Bets submissions - to report bets of the users

You can use 'Jackpots list' method to get list of eligible games and submit only bets that are related to these games.

JACKPOTS LIST

Request to Jackpots API for jackpots list

```
curl --request POST \
--url 'https://japi2.smartico.ai/api/jackpot/templates/v1' \
--header 'Authorization: 32b18199-229b-4081-8106-864d4f759ed6' \
--header 'Content-Type: application/json'
```

Example of response


```
{
    "jackpots": [
        {
            "related_games": [
                "trx1",
                "cleo1",
                "mon1",
                "bigp1",
                "wack1"
            ],
            "internal_name": "Iron Jackpot",
            "jp_currency": "USD",
            "jp_template_id": 11,
            "current_pot_amount": 100
        },
        {
            "related_games": [
                "cleo1"
            ],
            "internal_name": "Golden Jackpot",
            "jp_currency": "USD",
            "jp_template_id": 12,
            "current_pot_amount": 1.0032253848
        }
    ]
}
```


BETS SUBMISSIONS

Request to Jackpots API for Bets submissions

```
curl --request POST \
--url 'https://japi2.smartico.ai/api/jackpot/bets/v1' \
--header 'Authorization: 32b18199-229b-4081-8106-864d4f759ed6' \
--header 'Content-Type: application/json'
```

Every request to Jackpots API should contain the following fields as shown on the example below

Parameter	Description
req_id	Unique ID of request, used for logging purpose only
user_ext_id	ID of user in your system
ext_brand_id	ID of brand in your system
transaction_id	Unique ID of the bet
currency	The currency of the bet, e.g. EUR
bet_amount_real	Real money part of the bet, e.g. 1.55
bet_amount_bonus	Bonus money part of the bet, e.g. 0. Note that you can control if the bonus part of the bet should be included in the Jackpots calculations when configuring Jackpots in the Smartico BackOffice. Check for more details in the Jackpots documentation
game_ext_id	ID of the game in your system. Note that this ID should correspond to the ID of the game provided in the Games Catalog API

Parameter	Description
game_provider_ext_id	ID of the game provider in your system. Optional, can be used to define specific contribution per game provider

Example of Casino bets submissions to Jackpots API

```
{
    "req_id": "a78a3be6-d9bc-4959-a38c-4fe3f177aba8",
    "bets": [
        {
            "user_ext_id": "some_user_id_1",
            "ext_brand_id": "your_brand_ext_id",
            "transaction_id": "ab074ac3-881b-4ee9-b6ca-c8a9bb7e1752",
            "bet_amount_real": 1.5,
            "bet_amount_bonus": 0,
            "currency": "EUR",
            "game_ext_id": "game123",
            "game_provider_ext_id": "egt"
        },
        {
            "user_ext_id": "some_user_id_1",
            "ext_brand_id": "your_brand_ext_id",
            "transaction_id": "67e7b1ce-935f-4cc7-9634-139ecf4d4c92",
            "bet_amount_real": 0.5,
            "bet_amount_bonus": 0.5,
            "currency": "EUR",
            "game_ext_id": "game123",
            "game_provider_ext_id": "egt"
        },
        {
            "user_ext_id": "some_user_id_1",
            "ext_brand_id": "your_brand_ext_id",
            "transaction_id": "063a48d8-9356-4df3-92c6-e2497b9683f6",
            "bet_amount_real": 0,
            "bet_amount_bonus": 0.3,
            "currency": "EUR",
            "game_ext_id": "game55",
            "game_provider_ext_id": "pragmatic"
        }
    ]
}
```

ADDITIONAL APIS

GET USER STATE

There is an additional API to get the state of the user profile stored on Smartico side.
You can pass list of properties to retrieve in the 'properties' field of the payload as stated in the example.

Important
This API is additional service and not enabled by default, please contact your Account Manager to discuss your use case.
The API is limited to 30 requests per minute and 10 users per request.

Example of HTTP request

```
curl --request POST \
--url 'https://apis.smartico.ai/api/external/user_state/v2' \
--header 'Authorization: 32b18199-229b-4081-8106-864d4f759ed6' \
--header 'Content-Type: application/json'
```

Example TEST

```
{
    "eid": "ba34bad0-aec9-4180-998b-834892f9a4f6",
    "properties": [
        "core_registration_date",
        "core_wallet_currency",
        "ach_level_current",
        "core_public_tags",
        "user_all_campaigns",
        "core_user_last_login_date"
    ],
    "show_enum_values": true,
    "users": [
        {
            "user_ext_id": "some_user_id_1",
            "ext_brand_id": "your_brand_ext_id"
        }
    ]
}
```

UPDATE POINTS

The API can be used to update user points balance(s) on Smartico side.

Important
The API is limited to 30 requests per minute and 1000 users per request.

Example of HTTP request

```
curl --request POST \
--url 'https://apis.smartico.ai/api/external/user_points/v2' \
--header 'Authorization: 32b18199-229b-4081-8106-864d4f759ed6' \
--header 'Content-Type: application/json'
```

Example

TEST

```
{
    "eid": "54c77423-4fe7-4f36-8d11-dbd2e5f36236",
    "sync_level": true,
    "users": [
        {
            "user_ext_id": "test101",
            "ext_brand_id": "your_brand_ext_id",
            "ach_points_ever": 500,
            "ach_points_balance": 100,
            "ach_points_leaderboard": 50,
            "operation": "set"
        },
        {
            "user_ext_id": "test102",
            "ext_brand_id": "your_brand_ext_id",
            "ach_points_ever": 10,
            "ach_points_balance": 100,
            "ach_points_leaderboard": 50,
            "operation": "add"
        },
        {
            "user_ext_id": "test103",
            "ext_brand_id": "your_brand_ext_id",
            "ach_points_ever": 10,
            "ach_points_balance": 100,
            "ach_points_leaderboard": 50,
            "operation": "deduct"
        }
    ]
}
```

Example of the response:

```
{
    "req_id": "23decd82-2182-46e0-bfc9-eb70b54fdbb8",
    "pd": 34,
    "err_code": 0,
    "updated_users": [
        "test101",
        "test102"
    ],
    "errors": {
        "2a719c89-a8b1-437d-843f-e4ac6190c5e3": [
            "User was not found for a given label+brand/User test103 was not found in Smartico"
        ]
    },
    "empty": true
}
```

Handling duplicate requests

A deduplication check is performed based on the eid field in the request. If the same request is received a second time, the system will return an error message in the response:

```
{
    "req_id": "30e20920-a018-472b-a0fb-901acb8ed4e4",
    "eid": "5fd9dfee-80c0-494c-9eec-306818258121",
    "pd": 9,
```

```
    "err_code": 0,
    "updated_users": [],
    "errors": {
        "5fd9dfee-80c0-494c-9eec-306818258121": [
            "eid already processed 5fd9dfee-80c0-494c-9eec-306818258121 2283 create_date 123567890123"
        ]
    }
}
```

You can check the status of a previously submitted request—this is useful for implementing a retry mechanism.

Example of HTTP request

```
curl
--location 'https://apis[env_id].smartico.ai/api/external/user_points/v2?request_id=[id]'
--header 'Content-Type: application/json'
```

Recommended Retry Policy

In case of a connection timeout during an AddPoint API request, customers can implement the following retry strategy:

- Wait 3 minutes, then send a status check request using the same eid.
- If data is empty → the request was not received, and should be sent again.
- If data is present → the request was already processed, no need to retry.

Notes:

- As in the example, you can send batch updates, which is recommended way from performance perspective. Maximum number of users in the update - 1000
- For each user you need to specify: operation: 'set' | 'add' | 'deduct', ach_points_ever: int, ach_points_balance: int
- ach_points_ever - is reflecting points amount user collected ever, has the impact on the current level of the user
- ach_points_balance - current points balance of the user. Points can be spent in the store, mini-games, tournaments, etc.
- ach_points_leaderboard - points in all 3 types of leaderboards (daily, weekly, monthly)

UPDATE GEMS/DIAMONDS

The API can be used to update user gems/diamonds balance(s) on Smartico side.

Important
The API is limited to 30 requests per minute and 1000 users per request.

Example of HTTP request

```
curl --request POST \
--url 'https://apis.smartico.ai/api/external/user_points/gems_diamonds/v1' \
--header 'Authorization: 32b18199-229b-4081-8106-864d4f759ed6' \
--header 'Content-Type: application/json'
```

Example TEST

```
{
    "eid": "299ec2b8-47ee-472c-a734-a7c948c783aa",
    "sync_level": false,
    "users": [
        {
            "user_ext_id": "test101",
            "ext_brand_id": "your_brand_ext_id",
            "gems_balance": 50,
            "diamonds_balance": 100,
```

```
            "operation": "set"
        },
        {
            "user_ext_id": "test102",
            "ext_brand_id": "your_brand_ext_id",
            "gems_balance": 50,
            "operation": "add"
        },
        {
            "user_ext_id": "test103",
            "ext_brand_id": "your_brand_ext_id",
            "diamonds_balance": 100,
            "operation": "deduct"
        }
    ]
}
```

Example of the response:

```
{
    "req_id": "23decd82-2182-46e0-bfc9-eb70b54fdbb8",
    "pd": 34,
    "err_code": 0,
    "updated_users": [
        "test101",
        "test102"
    ],
    "errors": {
        "2a719c89-a8b1-437d-843f-e4ac6190c5e3": [
            "User was not found for a given label+brand/User test103 was not found in Smartico"
        ]
    },
    "empty": true
}
```

Handling duplicate requests

A deduplication check is performed based on the eid field in the request. If the same request is received a second time, the system will return an error message in the response:

```
{
    "req_id": "30e20920-a018-472b-a0fb-901acb8ed4e4",
    "eid": "5fd9dfee-80c0-494c-9eec-306818258121",
    "pd": 9,
    "err_code": 0,
    "updated_users": [],
    "errors": {
        "5fd9dfee-80c0-494c-9eec-306818258121": [
            "eid already processed 5fd9dfee-80c0-494c-9eec-306818258121 2283 create_date 123567890123"
        ]
    }
}
```

You can check the status of a previously submitted request—this is useful for implementing a retry mechanism.

Example of HTTP request

```
curl
--location 'https://apis[env_id].smartico.ai/api/external/user_points/gems_diamonds/v1?request_id=[id]'
--header 'Content-Type: application/json'
```

Recommended Retry Policy

In case of a connection timeout during an AddGemsAndDiamonds API request, customers can implement the following retry strategy:

- Wait 3 minutes, then send a status check request using the same eid.
- If data is empty → the request was not received, and should be sent again.
- If data is present → the request was already processed, no need to retry.

Notes:

- As in the example, you can send batch updates, which is recommended way from performance perspective. Maximum number of users in the update - 1000
- For each user you need to specify: operation: 'set' | 'add' | 'deduct', gems_balance: int, diamonds_balance: int
- gems_balance - current gems balance of the user
- diamonds_balance - current diamonds balance of the user

SUPPORTED PROPERTIES

All the properties list below can be updated with **update_profile** event in order to load historical data or to keep users in sync with Integrated system.
Also, most of these properties can be updated with domain specific events, e.g. **acc_last_deposit_amount** property can be updated with **acc_deposit_approved** event.

Smartico team recommends following approach:

- During initial integration to use **update_profile** event to synchronize state of all users into Smartico.
- Keep properties updated in realtime using domain specific events. E.g. when user makes deposit, bring values for all properties that are related to this event, like total deposit amount, total number of deposits, user balance after deposit etc.
- If some of the properties cannot be calculated at the moment of domain specific event, use **update_profile** event to deliver them when it's possible. E.g. GGR & NGR, that is usually complicated to calculated immidiatly at the moment of deposit, can be calculated once in a day and sent with update_profile event.
- You can run **update_profile** on all user base and pass needed properties if you have a bulk change of them on Product side or have indication that some of properties are out of sync with Smartico system.

CORE PROPERTIES

Property	Data type	Description	Stored

core_acc_pam_verified_dt
Date when account is verified in PAM
	timestamp	The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_acc_pam_verified_dt: 1757602862851`
		`}`
			-

core_account_status
Account status
	enumeration	Status of account in the smartico system. If the account is not ACTIVE, Smartico system will not execute any marketing activities for such an account. The default value for a newly created account is Active.
		`payload: {`
		`  core_account_status: 'ACTIVE'`
		`}`
		Possible values: ACTIVE, BLOCKED, SUSPENDED, BANNED, SELF_EXCLUDED, DEACTIVATED, PENDING, APPROVED
			449

core_aff_afp
AFP parameter from TAP
	string	AFP parameter from TheAffiliatePlatform in case was configured for synchronisation
		`payload: {`
		`  core_aff_afp: 'Some value'`
		`}`
			-

core_affiliate_id
Affiliate Id (numeric)
	numeric	Affiliate ID from affiliate system in case such ID is of type numeric
		`payload: {`
		`  core_affiliate_id: 123456`
		`}`
			6569

core_affiliate_parent_id
Parent affiliate Id (numeric)
	numeric	Parent Affiliate ID from affiliate system/TAP
		`payload: {`
		`  core_affiliate_parent_id: 123456`
		`}`
			-

core_affiliate_str
Affiliate Id (string)
	string	Affiliate ID from affiliate system in case such ID is of type string
		`payload: {`
		`  core_affiliate_str: 'x123456'`
		`}`
			-

Property	Data type	Description	Stored

core_btag
BTag
	string	BTag or "Banner tag", the unique ID used to track user from acquisition source
		`payload: {`
		`  core_btag: 'Some value'`
		`}`
			474

core_custom_prop1
Custom property 1
	string	Free text property that can be used for general purpose
		`payload: {`
		`  core_custom_prop1: 'Here could be any text value'`
		`}`
			6

core_custom_prop2
Custom property 2
	string	Free text property that can be used for general purpose
		`payload: {`
		`  core_custom_prop2: 'Here could be any text value'`
		`}`
			-

core_custom_prop3
Custom property 3
	string	Free text property that can be used for general purpose
		`payload: {`
		`  core_custom_prop3: 'Here could be any text value'`
		`}`
			-

core_custom_prop4
Custom property 4
	string	Free text property that can be used for general purpose
		`payload: {`
		`  core_custom_prop4: 'Here could be any text value'`
		`}`
			-

core_custom_prop5
Custom property 5
	string	Free text property that can be used for general purpose
		`payload: {`
		`  core_custom_prop5: 'Here could be any text value'`
		`}`
			-

core_custom_prop6
Custom property 6
	string	Free text property that can be used for general purpose
		`payload: {`
		`  core_custom_prop6: 'Here could be any text value'`
		`}`
			-

core_custom_prop7
Custom property 7
	string	Free text property that can be used for general purpose
		`payload: {`
		`  core_custom_prop7: 'Here could be any text value'`
		`}`
			-

core_custom_prop_dt1
Custom property date/time 1
	timestamp	Date/time property that can be used for general purpose
		The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_custom_prop_dt1: 1757602862853`
		`}`
			-

core_custom_prop_num1
Custom property, numeric 1
	numeric	Numberic property that can be used for general purpose
		`payload: {`
		`  core_custom_prop_num1: 123`
		`}`
			1

Property	Data type	Description	Stored

core_email_confirmed
Email confirmed by PAM
	boolean	Flag indicating that user has email confirmed
		`payload: {`
		`  core_email_confirmed: true`
		`}`
			-

core_email_valid_by_pam
Email valide according to the PAM
	boolean	Flag indicating that user has email valid according to PAM logic
		`payload: {`
		`  core_email_valid_by_pam: true`
		`}`
			-

core_exclude_from_marketing
Exclude from marketing
	boolean	Flag indicating that user should be excluded from all marketing communications. Doesn't have operational impact in Smartico system and should be used by the marketer in the segmentation
		`payload: {`
		`  core_exclude_from_marketing: true`
		`}`
			-

core_external_account_status
External Account status
	enumeration	Account status in the external system
		`payload: {`
		`  core_external_account_status: 'ACTIVE'`
		`}`
		Possible values: BLOCKED, ACTIVE
			1

core_external_mail_status
External mail status
	enumeration	Mail status in the external system
		`payload: {`
		`  core_external_mail_status: 'ACTIVE'`
		`}`
			-

core_external_markers
External markers
	enum array	User-level markers can be set from the external system via API. https://help.smartico.ai/welcome/products/general-concepts/user-markers-tags Important: property supports up to 1000 unique values.
		`payload: {`
		`  core_external_markers: ["Tag A","Tag B"]`
		`}`
			5

core_external_markers_2
External markers 2
	enum array	User-level markers can be set from the external system via API. https://help.smartico.ai/welcome/products/general-concepts/user-markers-tags Important: property supports up to 1000 unique values.
		`payload: {`
		`  core_external_markers_2: ["Tag A","Tag B"]`
		`}`
			-

core_external_phone_status
External phone status
	enumeration	Phone status in the external system
		`payload: {`
		`  core_external_phone_status: 'ACTIVE'`
		`}`
			-

core_external_segment
External segment
	enum array	Can be used to integrate with external segmentation system Important: property supports up to 1000 unique values.
		`payload: {`
		`  core_external_segment: ["Segment A","Segment B"]`
		`}`
			-

Property	Data type	Description	Stored

core_external_vip_level
External VIP level
	enumeration	VIP level of the user in the external system
		`payload: {`
		`  core_external_vip_level: 'Some value'`
		`}`
			-

core_is_acc_verified
Account verified
	boolean	Flag indicating that the user account is verified. The meaning and logic of verification is controlled on the operator side
		`payload: {`
		`  core_is_acc_verified: true`
		`}`
			-

core_is_email_disabled_by_platform
Is email opted out on integrated PAM
	boolean	User is opted out from mail communication on the Product side
		`payload: {`
		`  core_is_email_disabled_by_platform: true`
		`}`
			321

core_is_push_disabled_by_platform
Is push opted out on integrated PAM
	boolean	User is opted out from push communication on the Product side
		`payload: {`
		`  core_is_push_disabled_by_platform: true`
		`}`
			-

core_is_sms_disabled_by_platform
Is sms opted out on integrated PAM
	boolean	User is opted out from sms communication on the Product side
		`payload: {`
		`  core_is_sms_disabled_by_platform: true`
		`}`
			321

core_is_test_account
Is test account
	boolean	If account is marked as test or real. Test accounts are excluded by default from most of BI reports. Note: depending on product integration, this flag can be controlled by external system, changing it in Smartico CRM can be later overridden with next synchronization with external system.
		`payload: {`
		`  core_is_test_account: true`
		`}`
			138

core_kyc_completed_at
KYC Completed Date
	timestamp	Date when user completed KYC process
		The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_kyc_completed_at: 1757602862855`
		`}`
			-

core_kyc_started_at
KYC Started Date
	timestamp	Date when user started KYC process
		The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_kyc_started_at: 1757602862855`
		`}`
			-

core_kyc_status
KYC status
	enumeration	Status of the user in KYC process
		`payload: {`
		`  core_kyc_status: 'Some value'`
		`}`
			-

Property	Data type	Description	Stored

core_lifecycle
Lifecycle state
	enumeration	State of the lifecycle for user on integrated platform
		Important: property supports up to 20 unique values.
		`payload: {`
		`  core_lifecycle: 'DORMANT'`
		`}`
			-

core_lifecycle_update_dt
Lifecycle state update date
	timestamp	The date/time when state of the lifecycle for user is changed
		The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_lifecycle_update_dt: 1757602862856`
		`}`
			-

core_mail_domain
Mail domain
	string	Domain of the user mail address
		`payload: {`
		`  core_mail_domain: 'yahoo.com'`
		`}`
			-

core_main_device_type
Main device type
	enumeration	`payload: {`
		`  core_main_device_type: 'Some value'`
		`}`
			-

core_organization
Organization
	enumeration	Organization to which user belongs
		Important: property supports up to 100 unique values.
		`payload: {`
		`  core_organization: 'Google'`
		`}`
			-

core_pam_email_verified_dt
Date when email verified in PAM
	timestamp	The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_pam_email_verified_dt: 1757602862856`
		`}`
			-

core_pam_phone_verified_dt
Date when phone verified in PAM
	timestamp	The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_pam_phone_verified_dt: 1757602862856`
		`}`
			-

core_phone_confirmed
Phone confirmed by PAM
	boolean	Flag indicating that user phone is confirmed
		`payload: {`
		`  core_phone_confirmed: true`
		`}`
			-

core_pref_product
Preferred product by PAM
	enumeration	Type of product user prefers. E.g. sport, casino etc, reported from the integrated system
		Important: property supports up to 50 unique values.
		`payload: {`
		`  core_pref_product: 'SPORT'`
		`}`
			-

core_public_tags
JS markers
	enum array	JS markers of users - https://help.smartico.ai/welcome/products/general-concepts/user-markers-tags
		`payload: {`
		`  core_public_tags: ["Tag A","Tag B"]`
		`}`
			56

Property	Data type	Description	Stored

core_ref_code
Referral code
	string	The referral code that can be used by the meaning of the integrated platform
		`payload: {`
		`  core_ref_code: 'ABC123'`
		`}`
			-

core_reg_city
City
	string	City of the user
		`payload: {`
		`  core_reg_city: 'Paris'`
		`}`
			-

core_reg_state
State
	string	State of the user
		`payload: {`
		`  core_reg_state: 'Some value'`
		`}`
			-

core_registration_date
Registration date
	timestamp	The date/time when user is registered. When the property is populate, the system will trigger event 'Core: profile ready'. Note that this property cannot be changed once it is set.
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  core_registration_date: 1757602862857`
		`}`
			1233

core_registration_platform
Registration platform
	enumeration	Registration platform from where user was registered
		Important: property supports up to 100 unique values.
		`payload: {`
		`  core_registration_platform: 'DESKTOP'`
		`}`
			-

core_rfm_segment
RFM segment
	int	RFM (Recency, Frequency, Monetary) segmentation is a marketing technique that categorizes customers based on how recently they made a deposit (Recency), how often they make deposit (Frequency), and how much money they spend (Monetary). It helps identify valuable customer segments for targeted marketing strategies and retention efforts. https://help.smartico.ai/welcome/products/ai-models/rfm-analysis
		`payload: {`
		`  core_rfm_segment:`
		`}`
		Possible values: Champions, Loyal, Potential Loyalist, Promising, New Customers, Need Attention, About To Sleep, Hibernating Customers, At Risk, Losing But Engaged
			-

core_rfm_segment_prev
RFM segment (previous)
	int	Previous value of RFM segment, e.g. user migrated from "Loyal Customers" to "At Risk". This property will preserve his previous RFM segment type
		`payload: {`
		`  core_rfm_segment_prev:`
		`}`
		Possible values: Champions, Loyal, Potential Loyalist, Promising, New Customers, Need Attention, About To Sleep, Hibernating Customers, At Risk, Losing But Engaged
			-

core_risk_rank
Risk rank
	enumeration	Risk rank that can be used by the meaning specific for the platform
		Important: property supports up to 100 unique values.
		`payload: {`
		`  core_risk_rank: 'High'`
		`}`
			-

Property	Data type	Description	Stored

core_tags
User markers
	enum array	User-level markers can be set from the user profile or as part of the campaign. It can be used to build segments or inside the campaign to see if the user was marked. https://help.smartico.ai/welcome/products/general-concepts/user-markers-tags
		`payload: {`
		`  core_tags: ["Tag A","Tag B"]`
		`}`
			378

core_time_of_day_pref
Time of day preference
	enumeration	Important: property supports up to 20 unique values.
		`payload: {`
		`  core_time_of_day_pref: 'NIGHT'`
		`}`
			-

core_user_gender
Gender
	enumeration	`payload: {`
		`  core_user_gender: 'FEMALE'`
		`}`
		Possible values: MALE, FEMALE, F, M, UNKNOWN
			591

core_user_language
Language
	enumeration	Language of the user in ISO 639-1 format (EN,BG,IT). Note that for PT-BR, the value should be BR. For other casese, please consult with Smartico support
		`payload: {`
		`  core_user_language: 'EN'`
		`}`
			7228

core_user_last_login_date
Last login date
	timestamp	The date/time when user last logged in. Don't confuse with another propery - core_user_last_time_online. The login date is reported once per session, while online can be reported more often, e.g. when user does refresh of page or navigating between pages
		The passed value should be a timestamp in UTC with milliseconds precision.
		`payload: {`
		`  core_user_last_login_date: 1757602862858`
		`}`
			5339

core_user_type
User type
	enumeration	Type of user in the integrated platform
		Important: property supports up to 10 unique values.
		`payload: {`
		`  core_user_type: 'SPORT'`
		`}`
			-

core_username
Username
	string	Username or nickname
		`payload: {`
		`  core_username: 'john1980'`
		`}`
			475

core_utm_campaign
UTM Campaign
	string	UTM Campaign marker
		`payload: {`
		`  core_utm_campaign: 'promo_x'`
		`}`
			1

core_utm_content
UTM Content
	string	The utm content parameter is used to differentiate similar content or links within the same campaign. It's most commonly used in A/B testing and ad variations.
		`payload: {`
		`  core_utm_content: 'Green button'`
		`}`
			-

Property	Data type	Description	Stored

core_utm_medium
UTM Medium
	string	UTM Medium marker
		payload: {
		core_utm_medium: 'CPC, email, sms, banner etc'
		}
			-

core_utm_source
UTM Source
	string	UTM Source marker
		payload: {
		core_utm_source: 'fb'
		}
			1

core_wallet_currency
Wallet currency
	enumeration	Currency of the user wallet. Note that once set this value cannot be changed for user
		payload: {
		core_wallet_currency: 'EUR'
		}
		Possible values: UAH, BGN, EUR, GBP, USD, RON, MXN
			979

user_birthdate
Birthdate
	timestamp	Birth date of user
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		user_birthdate: 1757602862859
		}
			616

user_country
Registration country
	enumeration	2 symbols, user registration country, e.g. UK
		payload: {
		user_country: 'UK'
		}
			7104

user_email
Email
	string	In case Smartico Data Guard is setup, the mail is encrypted and cannot be accessed from Smartico CRM
		payload: {
		user_email: 'some@google.com'
		}
			111

user_email_status
Smartico email status (internal)
	enumeration	The status of the user's email is used to check if emails can be sent or if the email is valid. Possible statuses are None, New, Verified, Bounced, Blocked, Spam Reported, and Broken. Status is managed by the Smartico system based on the mail delivery responses from gateways. https://help.smartico.ai/welcome/products/crm-automation/communication-channels/opt-out-and-communication-statuses
		payload: {
		user_email_status: 'NEW'
		}
		Possible values: NEW, BROKEN, NONE, BLOCKED, VERIFIED, UNKNOWN, BOUNCED, DDD, SPAM REPORTED
			191

user_first_name
First name
	string	payload: {
		user_first_name: 'Christopher'
		}
			648

user_last_name
Last name
	string	payload: {
		user_last_name: 'Johnson'
		}
			628

Property	Data type	Description	Stored

user_phone
Phone
	string	Phone number for SMS/WhatsApp/Viber communication
		payload: {
		user_phone: '+3598812345678'
		}
			34

user_phone_country
Phone country
	enumeration	Country code based on user phone
		payload: {
		user_phone_country: 'Some value'
		}
			-

user_phone_prefix
Phone prefix
	string	Phone prefix in not-encoded way, can be used for segmentation
		payload: {
		user_phone_prefix: '+35988'
		}
			-

user_phone_status
Smartico phone status (internal)
	enumeration	None/Broken - The phone didn't pass Google phone validation, New - The phone passed Google verification, Verified - The phone was verified by click or other method, Failed to Deliver - We got failed to deliver SMS from Gateway, Not Mobile - Google have recognised that the number is not mobile one. https://help.smartico.ai/welcome/products/crm-automation/communication-channels/opt-out-and-communication-statuses
		payload: {
		user_phone_status: 'NEW'
		}
		Possible values: NEW, FAILED TO DELIVER, VERIFIED, NOT MOBILE, NONE/BROKEN, BROKEN, BROKENN
			5266

user_post_code
Post Code
	string	Postal code of user
		payload: {
		user_post_code: '1682'
		}
			474

ACCOUNTING PROPERTIES

Property	Data type	Description	Stored

acc_avg_deposit_amount
Average Deposit Amount
	monetary (numeric)	Average deposit amount in the player currency
		payload: {
		acc_avg_deposit_amount: 100.99
		}
			-

acc_avg_withdrawal_amount
Average Withdrawal Amount
	monetary (numeric)	Average withdrawal amount in the player currency
		payload: {
		acc_avg_withdrawal_amount: 100.99
		}
			-

acc_bonus_abuser
Bonus Abuser
	boolean	Flag indicating if the user is bonus abuser or blocked from getting bonuses
		payload: {
		acc_bonus_abuser: true
		}
			-

Property	Data type	Description	Stored
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	6
acc_bonus_score Bonus score	numeric	The bonus score calculated by PAM or by operator managed analytical function `payload: { acc_bonus_score: 1.255 }`	-
acc_bonus_to_dep_ratio Bonus to deposit ratio	numeric	`payload: { acc_bonus_to_dep_ratio: 0.25 }`	-
acc_documents_status Document Status	enumeration	Current status of documents in the KYC process. `payload: { acc_documents_status: 'Pending approval' }`	-
acc_ftd_amount FTD Amount	monetary (numeric)	Amount of the first deposit `payload: { acc_ftd_amount: 100.99 }`	-
acc_ftd_date FTD Date	timestamp	The date of the first deposit. The value is automatically set by Smartico based on the first transaction. If the Integrated product is able to deliver its own FTD date, it can be passed and will override the value defined by Smartico. The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { acc_ftd_date: 1757602862861 }`	1
acc_fwd_amount FWD Amount	monetary (numeric)	Amount of the first withdrawal `payload: { acc_fwd_amount: 100.99 }`	-
acc_fwd_date First approved withdrawal date	timestamp	The date of first approved withdrawal The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { acc_fwd_date: 1757602862861 }`	-
acc_gc_balance Gold coins balance	numeric	Gold coins balance `payload: { acc_gc_balance: 50.25 }`	1

Property	Data type	Description	Stored

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		payload: {
		acc_gc_bonus_balance: 50.25
		}
			1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			1

acc_last_30_days_avg_deposit_amount
Last 30 days average deposit amount
	monetary (numeric)	payload: {
		acc_last_30_days_avg_deposit_amount: 100.99
		}
			-

acc_last_30_days_bet_amount
Last 30 days bet amount
	monetary (numeric)	payload: {
		acc_last_30_days_bet_amount: 100.99
		}
			-

acc_last_30_days_bets_count
Last 30 days bets count
	numeric	payload: {
		acc_last_30_days_bets_count: 123
		}
			-

acc_last_30_days_deposit_amount
Last 30 days deposit amount
	monetary (numeric)	payload: {
		acc_last_30_days_deposit_amount: 100.99
		}
			-

acc_last_30_days_ggr
Last 30 days GGR
	monetary (numeric)	payload: {
		acc_last_30_days_ggr: 100.99
		}
			-

acc_last_30_days_ngr
Last 30 days NGR
	monetary (numeric)	payload: {
		acc_last_30_days_ngr: 100.99
		}
			-

acc_last_60_days_avg_deposit_amount
Last 60 days average deposit amount
	monetary (numeric)	payload: {
		acc_last_60_days_avg_deposit_amount: 100.99
		}
			-

Property	Data type	Description	Stored

acc_last_60_days_bet_amount
Last 60 days bet amount
	monetary (numeric)	payload: {
		acc_last_60_days_bet_amount: 100.99
		}
			-

acc_last_60_days_bets_count
Last 60 days bets count
	numeric	payload: {
		acc_last_60_days_bets_count: 123
		}
			-

acc_last_60_days_deposit_amount
Last 60 days deposit amount
	monetary (numeric)	payload: {
		acc_last_60_days_deposit_amount: 100.99
		}
			-

acc_last_60_days_ggr
Last 60 days GGR
	monetary (numeric)	payload: {
		acc_last_60_days_ggr: 100.99
		}
			-

acc_last_60_days_ngr
Last 60 days NGR
	monetary (numeric)	payload: {
		acc_last_60_days_ngr: 100.99
		}
			-

acc_last_7_days_avg_deposit_amount
Last 7 days average deposit amount
	monetary (numeric)	payload: {
		acc_last_7_days_avg_deposit_amount: 100.99
		}
			-

acc_last_7_days_bet_amount
Last 7 days bet amount
	monetary (numeric)	payload: {
		acc_last_7_days_bet_amount: 100.99
		}
			-

acc_last_7_days_bets_count
Last 7 days bets count
	numeric	payload: {
		acc_last_7_days_bets_count: 123
		}
			-

acc_last_7_days_deposit_amount
Last 7 days deposit amount
	monetary (numeric)	payload: {
		acc_last_7_days_deposit_amount: 100.99
		}
			-

acc_last_7_days_ggr
Last 7 days GGR
	monetary (numeric)	payload: {
		acc_last_7_days_ggr: 100.99
		}
			-

acc_last_7_days_ngr
Last 7 days NGR
	monetary (numeric)	payload: {
		acc_last_7_days_ngr: 100.99
		}
			-

acc_last_90_days_avg_deposit_amount
Last 90 days average deposit amount
	monetary (numeric)	payload: {
		acc_last_90_days_avg_deposit_amount: 100.99
		}
			-

acc_last_90_days_bet_amount
Last 90 days bet amount
	monetary (numeric)	payload: {
		acc_last_90_days_bet_amount: 100.99
		}
			-

Property	Data type	Description	Stored

acc_last_90_days_bets_count
Last 90 days bets count
	numeric	payload: {
		acc_last_90_days_bets_count: 123
		}
			-

acc_last_90_days_deposit_amount
Last 90 days deposit amount
	monetary (numeric)	payload: {
		acc_last_90_days_deposit_amount: 100.99
		}
			-

acc_last_90_days_ggr
Last 90 days GGR
	monetary (numeric)	payload: {
		acc_last_90_days_ggr: 100.99
		}
			-

acc_last_90_days_ngr
Last 90 days NGR
	monetary (numeric)	payload: {
		acc_last_90_days_ngr: 100.99
		}
			-

acc_last_active_currency
Last active currency
	enumeration	In case of multi-wallet accounts, should indicate currency of the current active wallet
		payload: {
		acc_last_active_currency: 'USD'
		}
		Possible values: USD
			2

acc_last_bonus_approved_code
Last approved bonus code
	string	Bonus type
		payload: {
		acc_last_bonus_approved_code: 'Some value'
		}
			-

acc_last_bonus_complete_date
Last complete bonus date
	timestamp	Date/time when the bonus was approved
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		acc_last_bonus_complete_date: 1757602862865
		}
			-

acc_last_bonus_date
Last approved bonus date
	timestamp	Date/time when the bonus was approved
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		acc_last_bonus_date: 1757602862865
		}
			4

acc_last_deposit_amount
Last Deposit Amount
	monetary (numeric)	Deposit amount in the player currency
		payload: {
		acc_last_deposit_amount: 100.99
		}
			25

acc_last_deposit_date
Last Deposit Date
	timestamp	The date of last deposit
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		acc_last_deposit_date: 1757602862866
		}
			24

Property	Data type	Description	Stored

acc_last_deposit_fail_date
Deposit last fail date
	timestamp	The date of last failed deposit
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		acc_last_deposit_fail_date: 1757602862866
		}
			-

acc_last_month_avg_deposit_amount
Last month average deposit amount
	monetary (numeric)	payload: {
		acc_last_month_avg_deposit_amount: 100.99
		}
			-

acc_last_month_bet_amount
Last month bet amount
	monetary (numeric)	payload: {
		acc_last_month_bet_amount: 100.99
		}
			-

acc_last_month_bets_count
Last month bets count
	numeric	payload: {
		acc_last_month_bets_count: 123
		}
			-

acc_last_month_deposit_amount
Last month deposit amount
	monetary (numeric)	payload: {
		acc_last_month_deposit_amount: 100.99
		}
			-

acc_last_month_ggr
Last month GGR
	monetary (numeric)	payload: {
		acc_last_month_ggr: 100.99
		}
			-

acc_last_month_ngr
Last month NGR
	monetary (numeric)	payload: {
		acc_last_month_ngr: 100.99
		}
			-

acc_last_week_avg_deposit_amount
Last week average deposit amount
	monetary (numeric)	payload: {
		acc_last_week_avg_deposit_amount: 100.99
		}
			-

acc_last_week_bet_amount
Last week bet amount
	monetary (numeric)	payload: {
		acc_last_week_bet_amount: 100.99
		}
			-

acc_last_week_bets_count
Last week bets count
	numeric	payload: {
		acc_last_week_bets_count: 123
		}
			-

acc_last_week_deposit_amount
Last week deposit amount
	monetary (numeric)	payload: {
		acc_last_week_deposit_amount: 100.99
		}
			-

acc_last_week_ggr
Last week GGR
	monetary (numeric)	payload: {
		acc_last_week_ggr: 100.99
		}
			-

Property	Data type	Description	Stored

acc_last_week_ngr
Last week NGR
	monetary (numeric)	payload: {
		acc_last_week_ngr: 100.99
		}
			-

acc_last_withdrawal_amount
Withdrawal Amount
	monetary (numeric)	Amount of with last withdrawal in the user currency
		payload: {
		acc_last_withdrawal_amount: 100.99
		}
			-

acc_last_withdrawal_date
Last approved withdrawal date
	timestamp	The date of last withdrawal
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		acc_last_withdrawal_date: 1757602862867
		}
			3

acc_median_deposit_amount
Median Deposit Amount
	monetary (numeric)	Median deposit amount in the player currency
		payload: {
		acc_median_deposit_amount: 100.99
		}
			-

acc_net_deposit_amount
Net Deposit Amount (lifetime)
	monetary (numeric)	payload: {
		acc_net_deposit_amount: 100.99
		}
			-

acc_pref_deposit_method
Preferred deposit method
	enumeration	Preferred deposit method of user. Calculated either by Integrated product or by Smartico analytical models
		payload: {
		acc_pref_deposit_method: 'Credit Card'
		}
			-

acc_pref_deposit_week_day
Preferred deposit week day
	enumeration	Preferred day of week for deposit. Calculated either by Integrated product or by Smartico analytical models
		payload: {
		acc_pref_deposit_week_day: 'Monday'
		}
			-

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		payload: {
		acc_sc_bonus_balance: 50.25
		}
			1

Property	Data type	Description	Stored

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		payload: {
		acc_sc_real_balance: 50.25
		}
			1

acc_total_approved_bonuse_amount
Total approved bonuses amount
	monetary (numeric)	An amount of approved bonuses that the user got ever. The value is automatically accumulated by Smartico based on the acc_bonus_approved transactions - every time a new acc_bonus_approved comes, Smartico will increase the amount. If the Integrated product is able to deliver its own acc_total_approved_bonuses_amount, it can be passed and will override the value defined by Smartico.
		payload: {
		acc_total_approved_bonuse_amount: 100.99
		}
			-

acc_total_approved_bonuses_count
Total approved bonuses count
	numeric	A number of approved bonuses that the user got ever. The value is automatically accumulated by Smartico based on the acc_bonus_approved transactions - every time a new acc_bonus_approved comes, Smartico will increase the count by 1. If the Integrated product is able to deliver its own acc_total_approved_bonuses_count, it can be passed and will override the value defined by Smartico.
		payload: {
		acc_total_approved_bonuses_count: 123
		}
			-

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		payload: {
		acc_total_balance: 50.25
		}
			1

acc_total_deposit_amount
Total Deposit Amount
	monetary (numeric)	Total deposit amount in the user currency. The value is automatically accumulated by Smartico based on the deposit transactions - every time a new deposit comes, Smartico will add it to the total deposit amount If the Integrated product is able to deliver its own calculation, it can be passed and will override the value defined by Smartico.
		payload: {
		acc_total_deposit_amount: 100.99
		}
			24

acc_total_deposit_count
Total Deposit Count
	numeric	Number of deposits that user did ever. The value is automatically accumulated by Smartico based on the deposit transactions - every time a new deposit comes, Smartico will increase deposit count by 1. If the Integrated product is able to deliver its own deposits count, it can be passed and will override the value defined by Smartico.
		payload: {
		acc_total_deposit_count: 123
		}
			24

acc_total_ggr
Total GGR
	monetary (numeric)	Gross Gaming Revenue in the user currency Usually bets minus wins amounts
		payload: {
		acc_total_ggr: 100.99
		}
			-

Property	Data type	Description	Stored
acc_total_lost_amount Total lost amount	monetary (numeric)	Total lost amount in the user currency `payload: { acc_total_lost_amount: 100.99 }`	-
acc_total_ngr Total NGR	monetary (numeric)	Net Gaming Revenue in the user currency `payload: { acc_total_ngr: 100.99 }`	-
acc_total_turnover Total turnover (wagering / bets)	monetary (numeric)	Total Turnover in the user currency `payload: { acc_total_turnover: 100.99 }`	-
acc_total_win_amount Total win amount	monetary (numeric)	Total win amount in the user currency `payload: { acc_total_win_amount: 100.99 }`	-
acc_total_withdrawal_amount Total Withdrawal Amount	monetary (numeric)	Total withdrawal amount in the user currency `payload: { acc_total_withdrawal_amount: 100.99 }`	-
acc_total_withdrawal_cancelled_amount Total Withdrawal Cancelled amount	monetary (numeric)	Total Withdrawal Cancelled amount `payload: { acc_total_withdrawal_cancelled_amount: 100.99 }`	-
acc_total_withdrawal_cancelled_cnt Total Withdrawal Cancelled count	numeric	Total count of cancelled withdrawals `payload: { acc_total_withdrawal_cancelled_cnt: 123 }`	-
acc_total_withdrawal_count Total Withdrawal Count	numeric	Number of withdrawals that user did `payload: { acc_total_withdrawal_count: 123 }`	-
acc_total_withdrawal_pending_amount Total Withdrawal Pending Amount	monetary (numeric)	`payload: { acc_total_withdrawal_pending_amount: 100.99 }`	-
acc_total_withdrawal_pending_cnt Total Withdrawal Pending count	numeric	Total count of pending withdrawals `payload: { acc_total_withdrawal_pending_cnt: 123 }`	-

Property	Data type	Description	Stored

acc_yesterday_avg_deposit_amount
Yesterday average deposit amount
	monetary (numeric)	payload: {
		acc_yesterday_avg_deposit_amount: 100.99
		}
			-

acc_yesterday_bet_amount
Yesterday bet amount
	monetary (numeric)	payload: {
		acc_yesterday_bet_amount: 100.99
		}
			-

acc_yesterday_bets_count
Yesterday bets count
	numeric	payload: {
		acc_yesterday_bets_count: 123
		}
			-

acc_yesterday_deposit_amount
Yesterday deposit amount
	monetary (numeric)	payload: {
		acc_yesterday_deposit_amount: 100.99
		}
			-

acc_yesterday_ggr
Yesterday GGR
	monetary (numeric)	payload: {
		acc_yesterday_ggr: 100.99
		}
			-

acc_yesterday_ngr
Yesterday NGR
	monetary (numeric)	payload: {
		acc_yesterday_ngr: 100.99
		}
			-

CASINO PROPERTIES

Property	Data type	Description	Stored

casino_avg_bet_amount_real
Average real money bet amount
	monetary (numeric)	Average lifetime bet amount of user in real money
		payload: {
		casino_avg_bet_amount_real: 10
		}
			1

casino_favorite_game
Favorite game
	enumeration	Favorite game of the user. The definition of a favorite game is managed by an Integrated product or by Smartico analytical models
		payload: {
		casino_favorite_game: 'Golden Slot'
		}
			-

casino_favorite_game2
Favorite game 2
	enumeration	Favorite game of the user. The definition of a favorite game is managed by an Integrated product or by Smartico analytical models
		payload: {
		casino_favorite_game2: 'Golden Slot'
		}
			-

casino_favorite_game_type
Favorite game type by PAM
	enumeration	Favorite game type of the user. The definition of a favorite game type is managed by an Integrated product or by Smartico analytical models
		payload: {
		casino_favorite_game_type: 'Golden Slot'
		}
			-

Property	Data type	Description	Stored

casino_last_30_days_bet_amount
Last 30 days bet amount
	monetary (numeric)	payload: {
		casino_last_30_days_bet_amount: 100.99
		}
			-

casino_last_30_days_bets_count
Last 30 days bets count
	numeric	payload: {
		casino_last_30_days_bets_count: 123
		}
			-

casino_last_30_days_ggr
Last 30 days GGR
	monetary (numeric)	payload: {
		casino_last_30_days_ggr: 100.99
		}
			-

casino_last_30_days_ngr
Last 30 days NGR
	monetary (numeric)	payload: {
		casino_last_30_days_ngr: 100.99
		}
			-

casino_last_30_days_win_amount
Last 30 days win amount
	monetary (numeric)	payload: {
		casino_last_30_days_win_amount: 100.99
		}
			-

casino_last_60_days_bet_amount
Last 60 days bet amount
	monetary (numeric)	payload: {
		casino_last_60_days_bet_amount: 100.99
		}
			-

casino_last_60_days_bets_count
Last 60 days bets count
	numeric	payload: {
		casino_last_60_days_bets_count: 123
		}
			-

casino_last_60_days_ggr
Last 60 days GGR
	monetary (numeric)	payload: {
		casino_last_60_days_ggr: 100.99
		}
			-

casino_last_60_days_ngr
Last 60 days NGR
	monetary (numeric)	payload: {
		casino_last_60_days_ngr: 100.99
		}
			-

casino_last_60_days_win_amount
Last 60 days win amount
	monetary (numeric)	payload: {
		casino_last_60_days_win_amount: 100.99
		}
			-

casino_last_7_days_bet_amount
Last 7 days bet amount
	monetary (numeric)	payload: {
		casino_last_7_days_bet_amount: 100.99
		}
			-

casino_last_7_days_bets_count
Last 7 days bets count
	numeric	payload: {
		casino_last_7_days_bets_count: 123
		}
			-

casino_last_7_days_ggr
Last 7 days GGR
	monetary (numeric)	payload: {
		casino_last_7_days_ggr: 100.99
		}
			-

Property	Data type	Description	Stored

casino_last_7_days_ngr
Last 7 days NGR
	monetary (numeric)	payload: {
		casino_last_7_days_ngr: 100.99
		}
			-

casino_last_7_days_win_amount
Last 7 days win amount
	monetary (numeric)	payload: {
		casino_last_7_days_win_amount: 100.99
		}
			-

casino_last_90_days_bet_amount
Last 90 days bet amount
	monetary (numeric)	payload: {
		casino_last_90_days_bet_amount: 100.99
		}
			-

casino_last_90_days_bets_count
Last 90 days bets count
	numeric	payload: {
		casino_last_90_days_bets_count: 123
		}
			-

casino_last_90_days_ggr
Last 90 days GGR
	monetary (numeric)	payload: {
		casino_last_90_days_ggr: 100.99
		}
			-

casino_last_90_days_ngr
Last 90 days NGR
	monetary (numeric)	payload: {
		casino_last_90_days_ngr: 100.99
		}
			-

casino_last_90_days_win_amount
Last 90 days win amount
	monetary (numeric)	payload: {
		casino_last_90_days_win_amount: 100.99
		}
			-

casino_last_bet_dt
Last Bet Date
	timestamp	Date/time of the bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		--- 	
		payload: {
		casino_last_bet_dt: 1757602862874
		}
			8

casino_last_bet_real_dt
Last Real Money Bet Date
	timestamp	Date/time of the bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		--- 	
		payload: {
		casino_last_bet_real_dt: 1757602862874
		}
			-

casino_last_month_bet_amount
Last month bet amount
	monetary (numeric)	payload: {
		casino_last_month_bet_amount: 100.99
		}
			-

casino_last_month_bets_count
Last month bets count
	numeric	payload: {
		casino_last_month_bets_count: 123
		}
			-

casino_last_month_ggr
Last month GGR
	monetary (numeric)	payload: {
		casino_last_month_ggr: 100.99
		}
			-

Property	Data type	Description	Stored

casino_last_month_ngr
Last month NGR
	monetary (numeric)	payload: {
		casino_last_month_ngr: 100.99
		}
			-

casino_last_month_win_amount
Last month win amount
	monetary (numeric)	payload: {
		casino_last_month_win_amount: 100.99
		}
			-

casino_last_week_bet_amount
Last week bet amount
	monetary (numeric)	payload: {
		casino_last_week_bet_amount: 100.99
		}
			-

casino_last_week_bets_count
Last week bets count
	numeric	payload: {
		casino_last_week_bets_count: 123
		}
			-

casino_last_week_ggr
Last week GGR
	monetary (numeric)	payload: {
		casino_last_week_ggr: 100.99
		}
			-

casino_last_week_ngr
Last week NGR
	monetary (numeric)	payload: {
		casino_last_week_ngr: 100.99
		}
			-

casino_last_week_win_amount
Last week win amount
	monetary (numeric)	payload: {
		casino_last_week_win_amount: 100.99
		}
			-

casino_total_bets_count
Total bets count
	numeric	Total count of casino bets
		payload: {
		casino_total_bets_count: 123
		}
			-

casino_total_free_bets_count
Total free bets count
	numeric	Total count of free casino bets
		payload: {
		casino_total_free_bets_count: 123
		}
			-

casino_total_ggr
Total / Lifetime GGR
	monetary (numeric)	payload: {
		casino_total_ggr: 100.99
		}
			-

casino_total_lost_amount
Total lost amount
	monetary (numeric)	Total lost amount in the user currency in Casino
		payload: {
		casino_total_lost_amount: 100.99
		}
			-

casino_total_ngr
Total / Lifetime NGR
	monetary (numeric)	payload: {
		casino_total_ngr: 100.99
		}
			-

Property	Data type	Description	Stored

casino_total_turnover
Total / Lifetime bet amount
	monetary (numeric)	Total Turnover / bet amount in the user currency
		payload: {
		casino_total_turnover: 100.99
		}
			-

casino_total_win_amount
Total / Lifetime win amount
	monetary (numeric)	Total win amount in the user currency in Casino
		payload: {
		casino_total_win_amount: 100.99
		}
			-

casino_yesterday_bet_amount
Yesterday bet amount
	monetary (numeric)	payload: {
		casino_yesterday_bet_amount: 100.99
		}
			-

casino_yesterday_bets_count
Yesterday bets count
	numeric	payload: {
		casino_yesterday_bets_count: 123
		}
			-

casino_yesterday_ggr
Yesterday GGR
	monetary (numeric)	payload: {
		casino_yesterday_ggr: 100.99
		}
			-

casino_yesterday_ngr
Yesterday NGR
	monetary (numeric)	payload: {
		casino_yesterday_ngr: 100.99
		}
			-

casino_yesterday_win_amount
Yesterday win amount
	monetary (numeric)	payload: {
		casino_yesterday_win_amount: 100.99
		}
			-

SPORT PROPERTIES

Property	Data type	Description	Stored

sport_first_bet_date
First Bet Date
	timestamp	Date of the first sport bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		sport_first_bet_date: 1757602862877
		}
			-

sport_last_30_days_bet_amount
Last 30 days bet amount
	monetary (numeric)	payload: {
		sport_last_30_days_bet_amount: 100.99
		}
			-

sport_last_30_days_bets_count
Last 30 days bets count
	numeric	payload: {
		sport_last_30_days_bets_count: 123
		}
			-

Property	Data type	Description	Stored

sport_last_30_days_ggr
Last 30 days GGR
	monetary (numeric)	payload: {
		sport_last_30_days_ggr: 100.99
		}
			-

sport_last_30_days_ngr
Last 30 days NGR
	monetary (numeric)	payload: {
		sport_last_30_days_ngr: 100.99
		}
			-

sport_last_30_days_win_amount
Last 30 days win amount
	monetary (numeric)	payload: {
		sport_last_30_days_win_amount: 100.99
		}
			-

sport_last_60_days_bet_amount
Last 60 days bet amount
	monetary (numeric)	payload: {
		sport_last_60_days_bet_amount: 100.99
		}
			-

sport_last_60_days_bets_count
Last 60 days bets count
	numeric	payload: {
		sport_last_60_days_bets_count: 123
		}
			-

sport_last_60_days_ggr
Last 60 days GGR
	monetary (numeric)	payload: {
		sport_last_60_days_ggr: 100.99
		}
			-

sport_last_60_days_ngr
Last 60 days NGR
	monetary (numeric)	payload: {
		sport_last_60_days_ngr: 100.99
		}
			-

sport_last_60_days_win_amount
Last 60 days win amount
	monetary (numeric)	payload: {
		sport_last_60_days_win_amount: 100.99
		}
			-

sport_last_7_days_bet_amount
Last 7 days bet amount
	monetary (numeric)	payload: {
		sport_last_7_days_bet_amount: 100.99
		}
			-

sport_last_7_days_bets_count
Last 7 days bets count
	numeric	payload: {
		sport_last_7_days_bets_count: 123
		}
			-

sport_last_7_days_ggr
Last 7 days GGR
	monetary (numeric)	payload: {
		sport_last_7_days_ggr: 100.99
		}
			-

sport_last_7_days_ngr
Last 7 days NGR
	monetary (numeric)	payload: {
		sport_last_7_days_ngr: 100.99
		}
			-

sport_last_7_days_win_amount
Last 7 days win amount
	monetary (numeric)	payload: {
		sport_last_7_days_win_amount: 100.99
		}
			-

Property	Data type	Description	Stored

sport_last_90_days_bet_amount
Last 90 days bet amount
	monetary (numeric)	payload: {
		sport_last_90_days_bet_amount: 100.99
		}
			-

sport_last_90_days_bets_count
Last 90 days bets count
	numeric	payload: {
		sport_last_90_days_bets_count: 123
		}
			-

sport_last_90_days_ggr
Last 90 days GGR
	monetary (numeric)	payload: {
		sport_last_90_days_ggr: 100.99
		}
			-

sport_last_90_days_ngr
Last 90 days NGR
	monetary (numeric)	payload: {
		sport_last_90_days_ngr: 100.99
		}
			-

sport_last_90_days_win_amount
Last 90 days win amount
	monetary (numeric)	payload: {
		sport_last_90_days_win_amount: 100.99
		}
			-

sport_last_bet_date
Last Bet Date
	timestamp	Date of the sport bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		--- 	
		payload: {
		sport_last_bet_date: 1757602862880
		}
			3

sport_last_bet_real_dt
Last Real Money Bet Date
	timestamp	Date/time of the bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		--- 	
		payload: {
		sport_last_bet_real_dt: 1757602862880
		}
			-

sport_last_month_bet_amount
Last month bet amount
	monetary (numeric)	payload: {
		sport_last_month_bet_amount: 100.99
		}
			-

sport_last_month_bets_count
Last month bets count
	numeric	payload: {
		sport_last_month_bets_count: 123
		}
			-

sport_last_month_ggr
Last month GGR
	monetary (numeric)	payload: {
		sport_last_month_ggr: 100.99
		}
			-

sport_last_month_ngr
Last month NGR
	monetary (numeric)	payload: {
		sport_last_month_ngr: 100.99
		}
			-

sport_last_month_win_amount
Last month win amount
	monetary (numeric)	payload: {
		sport_last_month_win_amount: 100.99
		}
			-

Property	Data type	Description	Stored

sport_last_week_bet_amount
Last week bet amount
	monetary (numeric)	payload: {
		sport_last_week_bet_amount: 100.99
		}
			-

sport_last_week_bets_count
Last week bets count
	numeric	payload: {
		sport_last_week_bets_count: 123
		}
			-

sport_last_week_ggr
Last week GGR
	monetary (numeric)	payload: {
		sport_last_week_ggr: 100.99
		}
			-

sport_last_week_ngr
Last week NGR
	monetary (numeric)	payload: {
		sport_last_week_ngr: 100.99
		}
			-

sport_last_week_win_amount
Last week win amount
	monetary (numeric)	payload: {
		sport_last_week_win_amount: 100.99
		}
			-

sport_total_ggr
Total GGR
	monetary (numeric)	payload: {
		sport_total_ggr: 100.99
		}
			-

sport_total_lost_amount
Total lost amount
	monetary (numeric)	Total lost amount in the user currency in Sport
		payload: {
		sport_total_lost_amount: 100.99
		}
			-

sport_total_ngr
Total NGR
	monetary (numeric)	payload: {
		sport_total_ngr: 100.99
		}
			-

sport_total_open_bets
Total / Lifetime bets count
	numeric	Number of sports betsa user has ever opened
		payload: {
		sport_total_open_bets: 3
		}
			-

sport_total_turnover
Total / Lifetime bet amount (wagering)
	monetary (numeric)	Total Turnover in the user currency
		payload: {
		sport_total_turnover: 100.99
		}
			-

sport_total_win_amount
Total win amount
	monetary (numeric)	Total win amount in the user currency in Sport
		payload: {
		sport_total_win_amount: 100.99
		}
			-

sport_yesterday_bet_amount
Yesterday bet amount
	monetary (numeric)	payload: {
		sport_yesterday_bet_amount: 100.99
		}
			-

Property	Data type	Description	Stored

sport_yesterday_bets_count
Yesterday bets count
	numeric	payload: {
		sport_yesterday_bets_count: 123
		}
			-

sport_yesterday_ggr
Yesterday GGR
	monetary (numeric)	payload: {
		sport_yesterday_ggr: 100.99
		}
			-

sport_yesterday_ngr
Yesterday NGR
	monetary (numeric)	payload: {
		sport_yesterday_ngr: 100.99
		}
			-

sport_yesterday_win_amount
Yesterday win amount
	monetary (numeric)	payload: {
		sport_yesterday_win_amount: 100.99
		}
			-

TELEGRAM PROPERTIES

Property	Data type	Description	Stored

core_telegram_chat_id
Telegram chat ID
	numeric	Populated when user is connected to the Telegram bot for personal messaging
		payload: {
		core_telegram_chat_id: 123
		}
			11

core_telegram_user_name
Telegram username / first name
	string	Populated when user is connected to the Telegram bot for personal messaging
		payload: {
		core_telegram_user_name: 'John1984'
		}
			-

DEMO PRODUCT PROPERTIES

Property	Data type	Description	Stored

demo_account_closed
Account Closed
	boolean	payload: {
		demo_account_closed: true
		}
			3

demo_bet_counter
Accumulated bets count (for points campaign)
	numeric	payload: {
		demo_bet_counter: 123
		}
			20

demo_regulation_type
Regulation type
	string	payload: {
		demo_regulation_type: 'Some value'
		}
			-

demo_user_segment
User segment
	enumeration	payload: {
		demo_user_segment: 'ACTIVE'
		}
		Possible values: ACTIVE
			220

CORE EVENTS

UPDATE PROFILE

A general structure that represents user profile.
Event type: update_profile

Example

TEST


```
{
    "eid": "af2c6eaa-bc6e-4fd7-9883-110b0adf50d3",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "update_profile",
    "payload": {
        "core_account_status": "ACTIVE",
        "user_first_name": "John",
        "user_last_name": "Wick",
        "core_user_gender": "M",
        "core_user_language": "UK",
        "user_country": "UA",
        "core_wallet_currency": "UAH",
        "user_birthdate": 1645669800000,
        "core_registration_date": 1757602862848,
        "core_external_markers": [
            "Test with API",
            "Marker X"
        ],
        "core_is_test_account": true,
        "core_affiliate_id": 777
    }
}
```

CLIENT ACTION

Custom client action that can be used to trigger different campaigns or automation rules. You can also pass custom data in the event payload, for example,

```
{
 "client_action": "complete_kyc",
 "link": "https://mysite.com/kyc"
}
```

And use the information from the event in the communication, e.g. {{event.link}}.

You can use this approach for other cases, like sending link with reset password, email verification etc.

Event type: client_action

Example TEST

```
{
    "eid": "8e8cb800-d68f-484b-9bee-ee0045cddbc4",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "client_action",
    "payload": {
        "client_action": "Some value",
        "demo_bet_counter": 123
    }
}
```

Payload field	Data type	Description	Required	Stored

client_action
Client action
	enumeration	Custom client action executed from web interface. Can be sent via _smartico.action('some action is here'). Note that you can send up to 500 different action types
		Important: property supports up to 500 unique values.
		--- 	
		`payload: {`
		`  client_action: 'Some value'`
		`}`
			-	Dynamic

demo_bet_counter
Accumulated bets count (for points campaign)
	numeric	`payload: {`
		`  demo_bet_counter: 123`
		`}`
			-	20

UPDATE FINANCIAL STATS

Event type: core_fin_stats_update

Example

TEST


```
{
    "eid": "006adde1-3a88-4082-9f97-2137af27f7dc",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "core_fin_stats_update",
    "payload": {
        "acc_total_ggr": 100.99,
        "acc_total_lost_amount": 100.99,
        "acc_total_ngr": 100.99,
        "acc_total_turnover": 100.99,
        "acc_total_win_amount": 100.99,
        "casino_total_bets_count": 123,
        "casino_total_free_bets_count": 123,
        "casino_total_ggr": 100.99,
        "casino_total_lost_amount": 100.99,
        "casino_total_ngr": 100.99,
        "casino_total_turnover": 100.99,
        "casino_total_win_amount": 100.99,
        "sport_total_ggr": 100.99,
        "sport_total_lost_amount": 100.99,
        "sport_total_ngr": 100.99,
        "sport_total_turnover": 100.99,
        "sport_total_win_amount": 100.99
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_total_ggr Total GGR	monetary (numeric)	Gross Gaming Revenue in the user currency Usually bets minus wins amounts `payload: { acc_total_ggr: 100.99 }`	-	
acc_total_lost_amount Total lost amount	monetary (numeric)	Total lost amount in the user currency `payload: { acc_total_lost_amount: 100.99 }`	-	
acc_total_ngr Total NGR	monetary (numeric)	Net Gaming Revenue in the user currency `payload: { acc_total_ngr: 100.99 }`	-	
acc_total_turnover Total turnover (wagering / bets)	monetary (numeric)	Total Turnover in the user currency `payload: { acc_total_turnover: 100.99 }`	-	
acc_total_win_amount Total win amount	monetary (numeric)	Total win amount in the user currency `payload: { acc_total_win_amount: 100.99 }`	-	

Payload field	Data type	Description	Required	Stored

casino_total_bets_count
Total bets count
	numeric	Total count of casino bets
		payload: {
		casino_total_bets_count: 123
		}
			-	

casino_total_free_bets_count
Total free bets count
	numeric	Total count of free casino bets
		payload: {
		casino_total_free_bets_count: 123
		}
			-	

casino_total_ggr
Total / Lifetime GGR
	monetary (numeric)	payload: {
		casino_total_ggr: 100.99
		}
			-	

casino_total_lost_amount
Total lost amount
	monetary (numeric)	Total lost amount in the user currency in Casino
		payload: {
		casino_total_lost_amount: 100.99
		}
			-	

casino_total_ngr
Total / Lifetime NGR
	monetary (numeric)	payload: {
		casino_total_ngr: 100.99
		}
			-	

casino_total_turnover
Total / Lifetime bet amount
	monetary (numeric)	Total Turnover / bet amount in the user currency
		payload: {
		casino_total_turnover: 100.99
		}
			-	

casino_total_win_amount
Total / Lifetime win amount
	monetary (numeric)	Total win amount in the user currency in Casino
		payload: {
		casino_total_win_amount: 100.99
		}
			-	

sport_total_ggr
Total GGR
	monetary (numeric)	payload: {
		sport_total_ggr: 100.99
		}
			-	

sport_total_lost_amount
Total lost amount
	monetary (numeric)	Total lost amount in the user currency in Sport
		payload: {
		sport_total_lost_amount: 100.99
		}
			-	

sport_total_ngr
Total NGR
	monetary (numeric)	payload: {
		sport_total_ngr: 100.99
		}
			-	

sport_total_turnover
Total / Lifetime bet amount (wagering)
	monetary (numeric)	Total Turnover in the user currency
		payload: {
		sport_total_turnover: 100.99
		}
			-	

Payload field	Data type	Description	Required	Stored

sport_total_win_amount
Total win amount
	monetary (numeric)	Total win amount in the user currency in Sport
		`payload: {`
		`  sport_total_win_amount: 100.99`
		`}`
			-	

ACCOUNTING EVENTS

WITHDRAWAL COMPLETED

Last transaction type in the withdrawal process. The usual flow is - requested -> approved -> completed. Note that this event DOES NOT increment acc_total_withdrawal_amount and acc_total_withdrawal_count properties. If you want to increment them, you need to send approved event.

Event type: acc_withdrawal_completed

Example TEST


```
{
    "eid": "dc26aa03-5303-4f45-aaf8-62b55494aed0",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_withdrawal_completed",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_transaction_id": "tid-485144",
        "acc_last_withdrawal_amount": 100.99,
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "acc_total_withdrawal_pending_amount": 100.99,
        "acc_utm_value": "fb54",
        "acc_wd_payment_method": "Credit Card",
        "acc_wd_payment_sub_method": "VISA",
        "acc_wd_source": "On-site",
        "acc_withdrawal_status_change_reason": "Some value"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic
acc_last_withdrawal_amount Withdrawal Amount	monetary (numeric)	Amount of with last withdrawal in the user currency `payload: { acc_last_withdrawal_amount: 100.99 }`	YES	

Payload field	Data type	Description	Required	Stored

acc_bonus_balance
Bonus balance
	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction
		payload: {
		acc_bonus_balance: 20.98
		}
			-	6

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		payload: {
		acc_gc_balance: 50.25
		}
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		payload: {
		acc_gc_bonus_balance: 50.25
		}
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		payload: {
		acc_sc_bonus_balance: 50.25
		}
			-	1

Payload field	Data type	Description	Required	Stored

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		`payload: {`
		`  acc_sc_real_balance: 50.25`
		`}`
			-	1

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		`payload: {`
		`  acc_total_balance: 50.25`
		`}`
			-	1

acc_total_withdrawal_pending_amount
Total Withdrawal Pending Amount
	monetary (numeric)	`payload: {`
		`  acc_total_withdrawal_pending_amount: 100.99`
		`}`
			-	

acc_utm_value
Utm value
	string	The UTM marker to highlight the marketing source of transaction
		`payload: {`
		`  acc_utm_value: 'fb54'`
		`}`
			-	Dynamic

acc_wd_payment_method
Withdrawal payment method
	enumeration	Name of used withdrawal method
		`payload: {`
		`  acc_wd_payment_method: 'Credit Card'`
		`}`
			-	Dynamic

acc_wd_payment_sub_method
Withdrawal payment sub-method
	enumeration	Name of used withdrawal sub-method
		`payload: {`
		`  acc_wd_payment_sub_method: 'VISA'`
		`}`
			-	Dynamic

acc_wd_source
Withdrawal source
	enumeration	Name of used withdrawal source (e.g. OnSite, Cash desk, etc)
		`payload: {`
		`  acc_wd_source: 'On-site'`
		`}`
			-	Dynamic

acc_withdrawal_status_change_reason
Withdrawal status change reason
	enumeration	Important: property supports up to 20 unique values.
		`payload: {`
		`  acc_withdrawal_status_change_reason: 'Some value'`
		`}`
			-	Dynamic

BONUS COMPLETED

Event type: acc_bonus_completed

Example

TEST


```
{
    "eid": "9e7bd763-8644-4b39-a502-d32b3e2ac44d",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_bonus_completed",
    "payload": {
        "acc_bonus_state": "Some value",
        "acc_last_bonus_amount": 100.99,
        "acc_last_bonus_code": "Some value",
        "acc_last_bonus_release_amount": 100.99,
        "acc_last_bonus_type": "Some value"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_bonus_state Bonus state	enumeration	State of the bonus, e.g. Active, Suspended, Deactivated, or any other value `payload: { acc_bonus_state: 'Some value' }`	-	Dynamic
acc_last_bonus_amount Bonus Amount	monetary (numeric)	Bonus amount in the player currency `payload: { acc_last_bonus_amount: 100.99 }`	-	Dynamic
acc_last_bonus_code Last Bonus Code	string	Bonus type `payload: { acc_last_bonus_code: 'Some value' }`	-	Dynamic
acc_last_bonus_release_amount Last Bonus Release Amount	monetary (numeric)	Last Bonus Release Amount `payload: { acc_last_bonus_release_amount: 100.99 }`	-	Dynamic
acc_last_bonus_type Bonus Type	enumeration	Bonus type `payload: { acc_last_bonus_type: 'Some value' }`	-	Dynamic

WITHDRAWAL REQUESTED

The initial transaction related to the withdrawal process. When the withdrawal is just requested by the end-user. Later it can be approved and completed or can be canceled.

Event type: acc_withdrawal_requested

Example TEST


```
{
    "eid": "1d1c63cb-0c21-417b-bcb9-623ad0968f8a",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_withdrawal_requested",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_transaction_id": "tid-485144",
        "acc_last_withdrawal_amount": 100.99,
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "acc_total_withdrawal_pending_amount": 100.99,
        "acc_utm_value": "fb54",
        "acc_wd_payment_method": "Credit Card",
        "acc_wd_payment_sub_method": "VISA",
        "acc_wd_source": "On-site",
        "acc_withdrawal_status_change_reason": "Some value"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic
acc_last_withdrawal_amount Withdrawal Amount	monetary (numeric)	Amount of with last withdrawal in the user currency `payload: { acc_last_withdrawal_amount: 100.99 }`	YES	
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	-	6

Payload field	Data type	Description	Required	Stored

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		payload: {
		acc_gc_balance: 50.25
		}
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		payload: {
		acc_gc_bonus_balance: 50.25
		}
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		payload: {
		acc_sc_bonus_balance: 50.25
		}
			-	1

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		payload: {
		acc_sc_real_balance: 50.25
		}
			-	1

Payload field	Data type	Description	Required	Stored

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		`payload: {`
		`  acc_total_balance: 50.25`
		`}`
			-	1

acc_total_withdrawal_pending_amount
Total Withdrawal Pending Amount
	monetary (numeric)	`payload: {`
		`  acc_total_withdrawal_pending_amount: 100.99`
		`}`
			-	

acc_utm_value
Utm value
	string	The UTM marker to highlight the marketing source of transaction
		`payload: {`
		`  acc_utm_value: 'fb54'`
		`}`
			-	Dynamic

acc_wd_payment_method
Withdrawal payment method
	enumeration	Name of used withdrawal method
		`payload: {`
		`  acc_wd_payment_method: 'Credit Card'`
		`}`
			-	Dynamic

acc_wd_payment_sub_method
Withdrawal payment sub-method
	enumeration	Name of used withdrawal sub-method
		`payload: {`
		`  acc_wd_payment_sub_method: 'VISA'`
		`}`
			-	Dynamic

acc_wd_source
Withdrawal source
	enumeration	Name of used withdrawal source (e.g. OnSite, Cash desk, etc)
		`payload: {`
		`  acc_wd_source: 'On-site'`
		`}`
			-	Dynamic

acc_withdrawal_status_change_reason
Withdrawal status change reason
	enumeration	Important: property supports up to 20 unique values.
		`payload: {`
		`  acc_withdrawal_status_change_reason: 'Some value'`
		`}`
			-	Dynamic

BONUS APPROVED

Event type: acc_bonus_approved

Example

TEST


```
{
    "eid": "e28eca82-c8bf-4ca3-a46a-2c3ef1fe31b7",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_bonus_approved",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_bonus_state": "Some value",
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_bonus_amount": 100.99,
        "acc_last_bonus_approved_code": "Some value",
        "acc_last_bonus_code": "Some value",
        "acc_last_bonus_date": 1757602862848,
        "acc_last_bonus_release_amount": 100.99,
        "acc_last_bonus_type": "Some value",
        "acc_last_transaction_id": "tid-485144",
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_last_bonus_amount": 123,
        "acc_sc_real_balance": 50.25,
        "acc_total_approved_bonuse_amount": 100.99,
        "acc_total_approved_bonuses_count": 123,
        "acc_total_balance": 50.25,
        "acc_utm_value": "fb54"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	-	6
acc_bonus_state Bonus state	enumeration	State of the bonus, e.g. Active, Suspended, Deactivated, or any other value `payload: { acc_bonus_state: 'Some value' }`	-	Dynamic

Payload field	Data type	Description	Required	Stored

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		payload: {
		acc_gc_balance: 50.25
		}
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		payload: {
		acc_gc_bonus_balance: 50.25
		}
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_last_bonus_amount
Bonus Amount
	monetary (numeric)	Bonus amount in the player currency
		payload: {
		acc_last_bonus_amount: 100.99
		}
			-	Dynamic

acc_last_bonus_approved_code
Last approved bonus code
	string	Bonus type
		payload: {
		acc_last_bonus_approved_code: 'Some value'
		}
			-	

acc_last_bonus_code
Last Bonus Code
	string	Bonus type
		payload: {
		acc_last_bonus_code: 'Some value'
		}
			-	Dynamic

acc_last_bonus_date
Last approved bonus date
	timestamp	Date/time when the bonus was approved
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		acc_last_bonus_date: 1757602862894
		}
			-	4

Payload field	Data type	Description	Required	Stored

acc_last_bonus_release_amount
Last Bonus Release Amount
	monetary (numeric)	Last Bonus Release Amount
		payload: {
		acc_last_bonus_release_amount: 100.99
		}
			-	Dynamic

acc_last_bonus_type
Bonus Type
	enumeration	Bonus type
		payload: {
		acc_last_bonus_type: 'Some value'
		}
			-	Dynamic

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		payload: {
		acc_sc_bonus_balance: 50.25
		}
			-	1

acc_sc_last_bonus_amount
Regular coins bonus amount
	numeric	payload: {
		acc_sc_last_bonus_amount: 123
		}
			-	Dynamic

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		payload: {
		acc_sc_real_balance: 50.25
		}
			-	1

acc_total_approved_bonuse_amount
Total approved bonuses amount
	monetary (numeric)	An amount of approved bonuses that the user got ever. The value is automatically accumulated by Smartico based on the acc_bonus_approved transactions - every time a new acc_bonus_approved comes, Smartico will increase the amount. If the Integrated product is able to deliver its own acc_total_approved_bonuses_amount, it can be passed and will override the value defined by Smartico.
		payload: {
		acc_total_approved_bonuse_amount: 100.99
		}
			-	

Payload field	Data type	Description	Required	Stored

acc_total_approved_bonuses_count
Total approved bonuses count
	numeric	A number of approved bonuses that the user got ever. The value is automatically accumulated by Smartico based on the acc_bonus_approved transactions - every time a new acc_bonus_approved comes, Smartico will increase the count by 1. If the Integrated product is able to deliver its own acc_total_approved_bonuses_count, it can be passed and will override the value defined by Smartico.
		`payload: {`
		`  acc_total_approved_bonuses_count: 123`
		`}`
			-	

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		`payload: {`
		`  acc_total_balance: 50.25`
		`}`
			-	1

acc_utm_value
Utm value
	string	The UTM marker to highlight the marketing source of transaction
		`payload: {`
		`  acc_utm_value: 'fb54'`
		`}`
			-	Dynamic

DEPOSIT APPROVED

Event type: acc_deposit_approved

Example

TEST


```
{
    "eid": "c070edd8-b740-47f8-988f-43fe0559b6c8",
    "event_date": 1757602862848,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_deposit_approved",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_deposit_cc_bin_code": "1234",
        "acc_ftd_date": 1757602862848,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_last_deposit_amount": 123,
        "acc_gc_real_balance": 50.25,
        "acc_is_first_deposit": "true",
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_active_currency": "USD",
        "acc_last_deposit_amount": 100.99,
        "acc_last_deposit_code": "PROMO1",
        "acc_last_deposit_date": 1757602862848,
        "acc_last_deposit_payment_method": "Credit Card",
        "acc_last_deposit_payment_submethod": "PayPal",
        "acc_last_deposit_platform": "MOBILE",
        "acc_last_deposit_source": "On-site",
        "acc_last_package_id": "Some value",
        "acc_last_transaction_id": "tid-485144",
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_last_deposit_amount": 123,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "acc_total_deposit_amount": 100.99,
        "acc_total_deposit_count": 123,
        "acc_utm_value": "fb54"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_deposit_amount Last Deposit Amount	monetary (numeric)	Deposit amount in the player currency `payload: { acc_last_deposit_amount: 100.99 }`	YES	25
acc_last_deposit_date Last Deposit Date	timestamp	The date of last deposit The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { acc_last_deposit_date: 1757602862896 }`	YES	24

Payload field	Data type	Description	Required	Stored

acc_last_transaction_id
Transaction Id
	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type
		`payload: {`
		`  acc_last_transaction_id: 'tid-485144'`
		`}`
			YES	Dynamic

acc_bonus_balance
Bonus balance
	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction
		`payload: {`
		`  acc_bonus_balance: 20.98`
		`}`
			-	6

acc_deposit_cc_bin_code
Deposit credit card bin code
	string	`payload: {`
		`  acc_deposit_cc_bin_code: '1234'`
		`}`
			-	Dynamic

acc_ftd_date
FTD Date
	timestamp	The date of the first deposit. The value is automatically set by Smartico based on the first transaction. If the Integrated product is able to deliver its own FTD date, it can be passed and will override the value defined by Smartico.
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  acc_ftd_date: 1757602862897`
		`}`
			-	1

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		`payload: {`
		`  acc_gc_balance: 50.25`
		`}`
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		`payload: {`
		`  acc_gc_bonus_balance: 50.25`
		`}`
			-	1

acc_gc_last_deposit_amount
Gold coins deposit amount
	numeric	`payload: {`
		`  acc_gc_last_deposit_amount: 123`
		`}`
			-	Dynamic

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		`payload: {`
		`  acc_gc_real_balance: 50.25`
		`}`
			-	1

acc_is_first_deposit
Is First Deposit
	boolean	Indicates if current deposit is a first one
		`payload: {`
		`  acc_is_first_deposit: true`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_last_active_currency
Last active currency
	enumeration	In case of multi-wallet accounts, should indicate currency of the current active wallet
		payload: {
		acc_last_active_currency: 'USD'
		}
		Possible values: USD
			-	2

acc_last_deposit_code
Deposit Code
	enumeration	Code used with last deposit
		payload: {
		acc_last_deposit_code: 'PROMO1'
		}
			-	Dynamic

acc_last_deposit_payment_method
Deposit payment method
	enumeration	Name of used deposit method
		payload: {
		acc_last_deposit_payment_method: 'Credit Card'
		}
			-	Dynamic

acc_last_deposit_payment_submethod
Deposit payment sub-method
	enumeration	Name of used deposit sub method
		payload: {
		acc_last_deposit_payment_submethod: 'PayPal'
		}
			-	Dynamic

acc_last_deposit_platform
Deposit platform
	enumeration	payload: {
		acc_last_deposit_platform: 'MOBILE'
		}
			-	Dynamic

acc_last_deposit_source
Deposit source
	enumeration	Source of the deposit
		payload: {
		acc_last_deposit_source: 'On-site'
		}
		Possible values: ON-SITE
			-	Dynamic

acc_last_package_id
Deposit package id
	string	payload: {
		acc_last_package_id: 'Some value'
		}
			-	Dynamic

Payload field	Data type	Description	Required	Stored

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		`payload: {`
		`  acc_real_balance: 50.25`
		`}`
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		`payload: {`
		`  acc_sc_balance: 50.25`
		`}`
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		`payload: {`
		`  acc_sc_bonus_balance: 50.25`
		`}`
			-	1

acc_sc_last_deposit_amount
Regular coins deposit amount
	numeric	`payload: {`
		`  acc_sc_last_deposit_amount: 123`
		`}`
			-	Dynamic

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		`payload: {`
		`  acc_sc_real_balance: 50.25`
		`}`
			-	1

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		`payload: {`
		`  acc_total_balance: 50.25`
		`}`
			-	1

acc_total_deposit_amount
Total Deposit Amount
	monetary (numeric)	Total deposit amount in the user currency. The value is automatically accumulated by Smartico based on the deposit transactions - every time a new deposit comes, Smartico will add it to the total deposit amount If the Integrated product is able to deliver its own calculation, it can be passed and will override the value defined by Smartico.
		`payload: {`
		`  acc_total_deposit_amount: 100.99`
		`}`
			-	24

acc_total_deposit_count
Total Deposit Count
	numeric	Number of deposits that user did ever. The value is automatically accumulated by Smartico based on the deposit transactions - every time a new deposit comes, Smartico will increase deposit count by 1. If the Integrated product is able to deliver its own deposits count, it can be passed and will override the value defined by Smartico.
		`payload: {`
		`  acc_total_deposit_count: 123`
		`}`
			-	24

Payload field	Data type	Description	Required	Stored

acc_utm_value
Utm value
	string	The UTM marker to highlight the marketing source of transaction
		`payload: {`
		`  acc_utm_value: 'fb54'`
		`}`
			-	Dynamic

WITHDRAWAL CANCELLED

The transaction represents canceling of a withdrawal request that was originally "requested" by the end-user.
Event type: acc_withdrawal_cancelled

Example TEST


```
{
    "eid": "8ab035c5-a8d5-4a57-b1fc-ba02296df269",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_withdrawal_cancelled",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_transaction_id": "tid-485144",
        "acc_last_withdrawal_amount": 100.99,
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "acc_total_withdrawal_cancelled_amount": 100.99,
        "acc_total_withdrawal_cancelled_cnt": 123,
        "acc_total_withdrawal_pending_amount": 100.99,
        "acc_utm_value": "fb54",
        "acc_wd_payment_method": "Credit Card",
        "acc_wd_payment_sub_method": "VISA",
        "acc_wd_source": "On-site",
        "acc_withdrawal_status_change_reason": "Some value"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic
acc_last_withdrawal_amount Withdrawal Amount	monetary (numeric)	Amount of with last withdrawal in the user currency `payload: { acc_last_withdrawal_amount: 100.99 }`	YES	
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	-	6

Payload field	Data type	Description	Required	Stored

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		payload: {
		acc_gc_balance: 50.25
		}
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		payload: {
		acc_gc_bonus_balance: 50.25
		}
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		payload: {
		acc_sc_bonus_balance: 50.25
		}
			-	1

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		payload: {
		acc_sc_real_balance: 50.25
		}
			-	1

Payload field	Data type	Description	Required	Stored

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		`payload: {`
		`  acc_total_balance: 50.25`
		`}`
			-	1

acc_total_withdrawal_cancelled_amount
Total Withdrawal Cancelled amount
	monetary (numeric)	Total Withdrawal Cancelled amount
		`payload: {`
		`  acc_total_withdrawal_cancelled_amount: 100.99`
		`}`
			-	

acc_total_withdrawal_cancelled_cnt
Total Withdrawal Cancelled count
	numeric	Total count of cancelled withdrawals
		`payload: {`
		`  acc_total_withdrawal_cancelled_cnt: 123`
		`}`
			-	

acc_total_withdrawal_pending_amount
Total Withdrawal Pending Amount
	monetary (numeric)	`payload: {`
		`  acc_total_withdrawal_pending_amount: 100.99`
		`}`
			-	

acc_utm_value
Utm value
	string	The UTM marker to highlight the marketing source of transaction
		`payload: {`
		`  acc_utm_value: 'fb54'`
		`}`
			-	Dynamic

acc_wd_payment_method
Withdrawal payment method
	enumeration	Name of used withdrawal method
		`payload: {`
		`  acc_wd_payment_method: 'Credit Card'`
		`}`
			-	Dynamic

acc_wd_payment_sub_method
Withdrawal payment sub-method
	enumeration	Name of used withdrawal sub-method
		`payload: {`
		`  acc_wd_payment_sub_method: 'VISA'`
		`}`
			-	Dynamic

acc_wd_source
Withdrawal source
	enumeration	Name of used withdrawal source (e.g. OnSite, Cash desk, etc)
		`payload: {`
		`  acc_wd_source: 'On-site'`
		`}`
			-	Dynamic

acc_withdrawal_status_change_reason
Withdrawal status change reason
	enumeration	Important: property supports up to 20 unique values.
		`payload: {`
		`  acc_withdrawal_status_change_reason: 'Some value'`
		`}`
			-	Dynamic

WITHDRAWAL APPROVED

2nd step in the withdrawal flow that usually built as: requested -> approved -> completed. Note that this event is incrementing acc_total_withdrawal_amount and acc_total_withdrawal_count properties

Event type: acc_withdrawal_approved

Example TEST


```
{
    "eid": "5fd925a7-43b6-4408-b29c-fbf021c0a0d1",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_withdrawal_approved",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_fwd_date": 1757602862849,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_transaction_id": "tid-485144",
        "acc_last_withdrawal_amount": 100.99,
        "acc_last_withdrawal_date": 1757602862849,
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_last_withdrawal_amount": 123,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "acc_total_withdrawal_amount": 100.99,
        "acc_total_withdrawal_count": 123,
        "acc_total_withdrawal_pending_amount": 100.99,
        "acc_utm_value": "fb54",
        "acc_wd_payment_method": "Credit Card",
        "acc_wd_payment_sub_method": "VISA",
        "acc_wd_source": "On-site",
        "acc_withdrawal_status_change_reason": "Some value"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic
acc_last_withdrawal_amount Withdrawal Amount	monetary (numeric)	Amount of with last withdrawal in the user currency `payload: { acc_last_withdrawal_amount: 100.99 }`	YES	
acc_last_withdrawal_date Last approved withdrawal date	timestamp	The date of last withdrawal The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { acc_last_withdrawal_date: 1757602862903 }`	YES	3

Payload field	Data type	Description	Required	Stored

acc_bonus_balance
Bonus balance
	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction
		payload: {
		acc_bonus_balance: 20.98
		}
			-	6

acc_fwd_date
First approved withdrawal date
	timestamp	The date of first approved withdrawal
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		payload: {
		acc_fwd_date: 1757602862903
		}
			-	

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		payload: {
		acc_gc_balance: 50.25
		}
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		payload: {
		acc_gc_bonus_balance: 50.25
		}
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			-	1

Payload field	Data type	Description	Required	Stored
acc_sc_bonus_balance Regular coins bonus balance	numeric	Regular coins bonus balance `payload: { acc_sc_bonus_balance: 50.25 }`	-	1
acc_sc_last_withdrawal_amount Regular coins withdrawal amount	numeric	`payload: { acc_sc_last_withdrawal_amount: 123 }`	-	Dynamic
acc_sc_real_balance Regular coins real balance	numeric	Regular coins real balance `payload: { acc_sc_real_balance: 50.25 }`	-	1
acc_total_balance Total balance	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction `payload: { acc_total_balance: 50.25 }`	-	1
acc_total_withdrawal_amount Total Withdrawal Amount	monetary (numeric)	Total withdrawal amount in the user currency `payload: { acc_total_withdrawal_amount: 100.99 }`	-	
acc_total_withdrawal_count Total Withdrawal Count	numeric	Number of withdrawals that user did `payload: { acc_total_withdrawal_count: 123 }`	-	
acc_total_withdrawal_pending_amount Total Withdrawal Pending Amount	monetary (numeric)	`payload: { acc_total_withdrawal_pending_amount: 100.99 }`	-	
acc_utm_value Utm value	string	The UTM marker to highlight the marketing source of transaction `payload: { acc_utm_value: 'fb54' }`	-	Dynamic
acc_wd_payment_method Withdrawal payment method	enumeration	Name of used withdrawal method `payload: { acc_wd_payment_method: 'Credit Card' }`	-	Dynamic
acc_wd_payment_sub_method Withdrawal payment sub-method	enumeration	Name of used withdrawal sub-method `payload: { acc_wd_payment_sub_method: 'VISA' }`	-	Dynamic

Payload field	Data type	Description	Required	Stored

acc_wd_source
Withdrawal source
	enumeration	Name of used withdrawal source (e.g. OnSite, Cash desk, etc)
		`payload: {`
		`  acc_wd_source: 'On-site'`
		`}`
			-	Dynamic

acc_withdrawal_status_change_reason
Withdrawal status change reason
	enumeration	Important: property supports up to 20 unique values.
		`payload: {`
		`  acc_withdrawal_status_change_reason: 'Some value'`
		`}`
			-	Dynamic

DEPOSIT CANCELLED

Event type: acc_deposit_cancelled

Example TEST


```
{
    "eid": "9e2c3dc7-7fb0-456d-b19f-768c457391e5",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_deposit_cancelled",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_deposit_code": "PROMO1",
        "acc_last_deposit_payment_method": "Credit Card",
        "acc_last_deposit_payment_submethod": "PayPal",
        "acc_last_deposit_platform": "MOBILE",
        "acc_last_deposit_source": "On-site",
        "acc_last_failed_deposit_amount": 100.99,
        "acc_last_transaction_id": "tid-485144",
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "acc_utm_value": "fb54"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	-	6
acc_gc_balance Gold coins balance	numeric	Gold coins balance `payload: { acc_gc_balance: 50.25 }`	-	1
acc_gc_bonus_balance Gold coins bonus balance	numeric	Gold coins bonus balance `payload: { acc_gc_bonus_balance: 50.25 }`	-	1

Payload field	Data type	Description	Required	Stored

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_last_deposit_code
Deposit Code
	enumeration	Code used with last deposit
		payload: {
		acc_last_deposit_code: 'PROMO1'
		}
			-	Dynamic

acc_last_deposit_payment_method
Deposit payment method
	enumeration	Name of used deposit method
		payload: {
		acc_last_deposit_payment_method: 'Credit Card'
		}
			-	Dynamic

acc_last_deposit_payment_submethod
Deposit payment sub-method
	enumeration	Name of used deposit sub method
		payload: {
		acc_last_deposit_payment_submethod: 'PayPal'
		}
			-	Dynamic

acc_last_deposit_platform
Deposit platform
	enumeration	payload: {
		acc_last_deposit_platform: 'MOBILE'
		}
			-	Dynamic

acc_last_deposit_source
Deposit source
	enumeration	Source of the deposit
		payload: {
		acc_last_deposit_source: 'On-site'
		}
		Possible values: ON-SITE
			-	Dynamic

acc_last_failed_deposit_amount
Last Failed Deposit Amount
	monetary (numeric)	Last Failed Deposit amount in the player currency
		payload: {
		acc_last_failed_deposit_amount: 100.99
		}
			-	Dynamic

Payload field	Data type	Description	Required	Stored

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		`payload: {`
		`  acc_real_balance: 50.25`
		`}`
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		`payload: {`
		`  acc_sc_balance: 50.25`
		`}`
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		`payload: {`
		`  acc_sc_bonus_balance: 50.25`
		`}`
			-	1

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		`payload: {`
		`  acc_sc_real_balance: 50.25`
		`}`
			-	1

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		`payload: {`
		`  acc_total_balance: 50.25`
		`}`
			-	1

acc_utm_value
Utm value
	string	The UTM marker to highlight the marketing source of transaction
		`payload: {`
		`  acc_utm_value: 'fb54'`
		`}`
			-	Dynamic

BONUS CANCELLED

Event type: acc_bonus_cancelled

Example

TEST


```
{
    "eid": "219e975c-a198-433c-a104-5d29cea9abe2",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_bonus_cancelled",
    "payload": {
        "acc_bonus_state": "Some value",
        "acc_last_bonus_amount": 100.99,
        "acc_last_bonus_cancel_reason": "Some value",
        "acc_last_bonus_code": "Some value",
        "acc_last_bonus_type": "Some value"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_bonus_state Bonus state	enumeration	State of the bonus, e.g. Active, Suspended, Deactivated, or any other value `payload: { acc_bonus_state: 'Some value' }`	-	Dynamic
acc_last_bonus_amount Bonus Amount	monetary (numeric)	Bonus amount in the player currency `payload: { acc_last_bonus_amount: 100.99 }`	-	Dynamic
acc_last_bonus_cancel_reason Bonus Cancel Reason	enumeration	Bonus type `payload: { acc_last_bonus_cancel_reason: 'Some value' }`	-	Dynamic
acc_last_bonus_code Last Bonus Code	string	Bonus type `payload: { acc_last_bonus_code: 'Some value' }`	-	Dynamic
acc_last_bonus_type Bonus Type	enumeration	Bonus type `payload: { acc_last_bonus_type: 'Some value' }`	-	Dynamic

FINANCIAL DELTA

The event can be used to send delta changes for different financial parameters. For example, you can send daily changes of user GGR and use it later in the behavioral segments

Event type: acc_financial_delta

Example TEST


```
{
    "eid": "73f0d980-ba23-4121-aa7c-33c0697fcb8f",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_financial_delta",
    "payload": {
        "acc_delta_net_deposit": 100.99
    }
}
```


Payload field	Data type	Description	Required	Stored

acc_delta_net_deposit
Delta of net deposit
	monetary (numeric)	Delta of net-deposit since the last reporting. Should be in the user wallet currency
		payload: {
		acc_delta_net_deposit: 100.99
		}
			-	Dynamic

ACCOUNTING METRICS UPDATE

This is an optional event that can be used to update accounting properties that are not directly connected to general events
For example, you may have a week ETL process that calculates metrics for each player, and you want to use these metrics in Smartico for segmentation.

Event type: acc_info_update

Example TEST


```
{
    "eid": "2ca4db71-e41b-428c-95cb-39ff37aee162",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_info_update",
    "payload": {}
}
```

BONUS FAILED

Event type: acc_bonus_failed

Example

TEST

```
{
    "eid": "c9e866e9-7502-44bd-bd1f-3cb0af8db1eb",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_bonus_failed",
    "payload": {
        "acc_bonus_state": "Some value",
        "acc_last_bonus_amount": 100.99,
        "acc_last_bonus_code": "Some value",
        "acc_last_bonus_failed_reason": "Some value"
    }
}
```

Payload field	Data type	Description	Required	Stored
acc_bonus_state Bonus state	enumeration	State of the bonus, e.g. Active, Suspended, Deactivated, or any other value `payload: { acc_bonus_state: 'Some value' }`	-	Dynamic
acc_last_bonus_amount Bonus Amount	monetary (numeric)	Bonus amount in the player currency `payload: { acc_last_bonus_amount: 100.99 }`	-	Dynamic
acc_last_bonus_code Last Bonus Code	string	Bonus type `payload: { acc_last_bonus_code: 'Some value' }`	-	Dynamic
acc_last_bonus_failed_reason Last Bonus failed reason	enumeration	Bonus type Important: property supports up to 20 unique values. `payload: { acc_last_bonus_failed_reason: 'Some value' }`	-	Dynamic

DEPOSIT ATTEMPT

Event type: acc_deposit_attempt

Example

TEST

```
{
    "eid": "7d0a7111-eb6e-4c76-af17-25095cd5fa2c",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_deposit_attempt",
    "payload": {
        "acc_last_deposit_attempt_amount": 100.99,
        "acc_last_transaction_id": "tid-485144"
    }
}
```

Payload field	Data type	Description	Required	Stored
acc_last_deposit_attempt_amount Last Deposit Attempt Amount	monetary (numeric)	Deposit attempt amount in the player currency `payload: { acc_last_deposit_attempt_amount: 100.99 }`	YES	Dynamic
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic

DEPOSIT FAILED

Event type: acc_deposit_failed

Example TEST


```
{
    "eid": "4b74fdfa-4968-42ef-a935-2c566a223afc",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "acc_deposit_failed",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_deposit_code": "PROMO1",
        "acc_last_deposit_fail_date": 1757602862849,
        "acc_last_deposit_payment_method": "Credit Card",
        "acc_last_deposit_payment_submethod": "PayPal",
        "acc_last_deposit_platform": "MOBILE",
        "acc_last_deposit_source": "On-site",
        "acc_last_failed_deposit_amount": 100.99,
        "acc_last_failed_deposit_code": "Some value",
        "acc_last_transaction_id": "tid-485144",
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "acc_utm_value": "fb54"
    }
}
```


Payload field	Data type	Description	Required	Stored
acc_last_deposit_fail_date Deposit last fail date	timestamp	The date of last failed deposit The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { acc_last_deposit_fail_date: 1757602862911 }`	YES	
acc_last_transaction_id Transaction Id	string	Transaction ID of last money related operation (deposit, withdrawal, bonus etc). Can be a free text string, unique in the context of specific operation type `payload: { acc_last_transaction_id: 'tid-485144' }`	YES	Dynamic
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	-	6

Payload field	Data type	Description	Required	Stored

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		payload: {
		acc_gc_balance: 50.25
		}
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		payload: {
		acc_gc_bonus_balance: 50.25
		}
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		payload: {
		acc_gc_real_balance: 50.25
		}
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_bonus: 50.25
		}
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_last_deposit_code
Deposit Code
	enumeration	Code used with last deposit
		payload: {
		acc_last_deposit_code: 'PROMO1'
		}
			-	Dynamic

acc_last_deposit_payment_method
Deposit payment method
	enumeration	Name of used deposit method
		payload: {
		acc_last_deposit_payment_method: 'Credit Card'
		}
			-	Dynamic

acc_last_deposit_payment_submethod
Deposit payment sub-method
	enumeration	Name of used deposit sub method
		payload: {
		acc_last_deposit_payment_submethod: 'PayPal'
		}
			-	Dynamic

acc_last_deposit_platform
Deposit platform
	enumeration	payload: {
		acc_last_deposit_platform: 'MOBILE'
		}
			-	Dynamic

acc_last_deposit_source
Deposit source
	enumeration	Source of the deposit
		payload: {
		acc_last_deposit_source: 'On-site'
		}
		Possible values: ON-SITE
			-	Dynamic

Payload field	Data type	Description	Required	Stored

acc_last_failed_deposit_amount
Last Failed Deposit Amount
	monetary (numeric)	Last Failed Deposit amount in the player currency
		`payload: {`
		`  acc_last_failed_deposit_amount: 100.99`
		`}`
			-	Dynamic

acc_last_failed_deposit_code
Last failed deposit code / reason
	enumeration	`payload: {`
		`  acc_last_failed_deposit_code: 'Some value'`
		`}`
			-	Dynamic

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		`payload: {`
		`  acc_real_balance: 50.25`
		`}`
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		`payload: {`
		`  acc_sc_balance: 50.25`
		`}`
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		`payload: {`
		`  acc_sc_bonus_balance: 50.25`
		`}`
			-	1

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		`payload: {`
		`  acc_sc_real_balance: 50.25`
		`}`
			-	1

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		`payload: {`
		`  acc_total_balance: 50.25`
		`}`
			-	1

acc_utm_value
Utm value
	string	The UTM marker to highlight the marketing source of transaction
		`payload: {`
		`  acc_utm_value: 'fb54'`
		`}`
			-	Dynamic

CASINO EVENTS

BET

It's important for the 'bet' event to have casino_last_bet_id the same as the 'win' event.
Smartico has the logic that connects 'bet' and corresponding 'win' events and transforms them into 'bet-win' event that will have the original bet amount taken from the 'bet' and 'win' amount taken from the win.

Note that by default, Smartico will expect 'win' event to be reported within one minute after 'bet'. If the 'win' was not reporting, Smartico will assume that the bet was losing and will create 'bet-win' event with 0 win amount. The waiting period can be adjusted.

Also, the win amounts will be summed up if multiple wins are reported during the waiting period.

Event type: casino_bet

Example TEST


```
{
    "eid": "5bb916c5-1421-4bf6-87c3-0df67da0e598",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "casino_bet",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_active_currency": "USD",
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "casino_game_ext_id": "Some value",
        "casino_game_provider_ext_id": "Some value",
        "casino_game_type_ext_id": "Some value",
        "casino_gc_bet_amount": 4,
        "casino_is_free_bet": "true",
        "casino_last_bet_amount": 5,
        "casino_last_bet_amount_bonus": 1,
        "casino_last_bet_amount_real": 4,
        "casino_last_bet_dt": 1757602862849,
        "casino_last_bet_game_name": "Super Slot",
        "casino_last_bet_game_provider": "Playtech",
        "casino_last_bet_game_type": "Slots",
        "casino_last_bet_game_type_arr": [
            "Value 1",
            "Value 2"
        ],
        "casino_last_bet_game_vendor": "BTG",
        "casino_last_bet_id": "betid-123552355",
        "casino_last_bet_platform": "WEB",
        "casino_last_bet_real_dt": 1757602862849,
        "casino_last_bet_status": "Rollback",
        "casino_sc_bet_amount": 4
    }
}
```

Payload field	Data type	Description	Required	Stored
casino_last_bet_amount Bet Amount	monetary (numeric)	Bet amount in the user currency (total of bonus and real) `payload: { casino_last_bet_amount: 5 }`	YES	Dynamic
casino_last_bet_dt Last Bet Date	timestamp	Date/time of the bet The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { casino_last_bet_dt: 1757602862914 }`	YES	8
casino_last_bet_game_name Bet Game Name	enumeration	Name of the game, e.g. "Super Slot" `payload: { casino_last_bet_game_name: 'Super Slot' }`	YES	Dynamic
casino_last_bet_id Bet id	string	Unique ID of the bet `payload: { casino_last_bet_id: 'betid-123552355' }`	YES	Dynamic
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	-	6
acc_gc_balance Gold coins balance	numeric	Gold coins balance `payload: { acc_gc_balance: 50.25 }`	-	1
acc_gc_bonus_balance Gold coins bonus balance	numeric	Gold coins bonus balance `payload: { acc_gc_bonus_balance: 50.25 }`	-	1
acc_gc_real_balance Gold coins balance (real)	numeric	Gold coins real balance `payload: { acc_gc_real_balance: 50.25 }`	-	1
acc_land_balance_bonus Bonus balance, landbase	monetary (numeric)	`payload: { acc_land_balance_bonus: 50.25 }`	-	1
acc_land_balance_real Real balance, landbase	monetary (numeric)	`payload: { acc_land_balance_real: 50.25 }`	-	1

Payload field	Data type	Description	Required	Stored

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_last_active_currency
Last active currency
	enumeration	In case of multi-wallet accounts, should indicate currency of the current active wallet
		payload: {
		acc_last_active_currency: 'USD'
		}
		Possible values: USD
			-	2

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		payload: {
		acc_sc_bonus_balance: 50.25
		}
			-	1

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		payload: {
		acc_sc_real_balance: 50.25
		}
			-	1

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		payload: {
		acc_total_balance: 50.25
		}
			-	1

casino_game_ext_id
Game external ID (from catalog)
	string	ID of the Game on the operator side
		payload: {
		casino_game_ext_id: 'Some value'
		}
			-	Dynamic

casino_game_provider_ext_id
Game provider external ID (from catalog)
	string	ID of the Game provider on the operator side
		payload: {
		casino_game_provider_ext_id: 'Some value'
		}
			-	Dynamic

Payload field	Data type	Description	Required	Stored

casino_game_type_ext_id
Game type external ID (from catalog)
	string	ID of the Game provider on the operator side
		`payload: {`
		`  casino_game_type_ext_id: 'Some value'`
		`}`
			-	Dynamic

casino_gc_bet_amount
Casino bet amount (gold coins)
	numeric	Bet amount in the sweepstake gold coins
		`payload: {`
		`  casino_gc_bet_amount: 4`
		`}`
			-	Dynamic

casino_is_free_bet
Is free bet
	boolean	Indicator if it was free bet
		`payload: {`
		`  casino_is_free_bet: true`
		`}`
			-	Dynamic

casino_last_bet_amount_bonus
Bet Amount (Bonus)
	monetary (numeric)	Bet amount in the user currency, bonus money.
		`payload: {`
		`  casino_last_bet_amount_bonus: 1`
		`}`
			-	Dynamic

casino_last_bet_amount_real
Bet Amount (Real)
	monetary (numeric)	Bet amount in the user currency, real money
		`payload: {`
		`  casino_last_bet_amount_real: 4`
		`}`
			-	Dynamic

casino_last_bet_game_provider
Bet Game Provider
	enumeration	Game provider, e.g. "Playtech"
		`payload: {`
		`  casino_last_bet_game_provider: 'Playtech'`
		`}`
			-	Dynamic

casino_last_bet_game_type
Bet Game Type
	enumeration	Game type, e.g. "Slots"
		`payload: {`
		`  casino_last_bet_game_type: 'Slots'`
		`}`
			-	Dynamic

casino_last_bet_game_type_arr
Bet Game Types
	enum array	Set of game types, e.g. "Slots" & "Super slots". Used when platform has multiple types assigned to the game
		`payload: {`
		`  casino_last_bet_game_type_arr: ["Value 1","Value 2"]`
		`}`
		Possible values: VALUE 1, VALUE 2
			-	Dynamic

casino_last_bet_game_vendor
Bet Game Vendor
	enumeration	Game vendor, e.g. "BTG"
		`payload: {`
		`  casino_last_bet_game_vendor: 'BTG'`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored

casino_last_bet_platform
Bet platform
	enumeration	The platform type from which the bet was done, e.g. WEB, MOBILE, APP etc.
		Important: property supports up to 10 unique values.
		`payload: {`
		`  casino_last_bet_platform: 'WEB'`
		`}`
		Possible values: WEB
			-	Dynamic

casino_last_bet_real_dt
Last Real Money Bet Date
	timestamp	Date/time of the bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  casino_last_bet_real_dt: 1757602862917`
		`}`
			-	

casino_last_bet_status
Bet Status
	enumeration	Optional status of the bet, for example can be used to indicate that win was marked as "rollback"
		`payload: {`
		`  casino_last_bet_status: 'Rollback'`
		`}`
		Possible values: ROLLBACK, WIN
			-	Dynamic

casino_sc_bet_amount
Casino bet amount (regular coins)
	numeric	Bet amount in the sweepstake coins (regular coins)
		`payload: {`
		`  casino_sc_bet_amount: 4`
		`}`
			-	Dynamic

WIN

See the 'bet' event explanation..

Event type: casino_win

Example TEST


```
{
    "eid": "1f5c89a4-7591-4339-8984-40ef4fe762bd",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "casino_win",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_active_currency": "USD",
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "casino_avg_bet_amount_real": 10,
        "casino_game_ext_id": "Some value",
        "casino_game_provider_ext_id": "Some value",
        "casino_game_type_ext_id": "Some value",
        "casino_gc_bet_amount": 4,
        "casino_gc_win_amount": 4,
        "casino_is_free_bet": "true",
        "casino_last_bet_amount": 5,
        "casino_last_bet_amount_bonus": 1,
        "casino_last_bet_amount_real": 4,
        "casino_last_bet_game_name": "Super Slot",
        "casino_last_bet_game_provider": "Playtech",
        "casino_last_bet_game_type": "Slots",
        "casino_last_bet_game_type_arr": [
            "Value 1",
            "Value 2"
        ],
        "casino_last_bet_game_vendor": "BTG",
        "casino_last_bet_id": "betid-123552355",
        "casino_last_bet_platform": "WEB",
        "casino_last_bet_profit": 8,
        "casino_last_bet_profit_total": 8,
        "casino_last_bet_status": "Rollback",
        "casino_last_win_amount": 10,
        "casino_last_win_amount_bonus": 8,
        "casino_last_win_amount_real": 8,
        "casino_sc_bet_amount": 4,
        "casino_sc_win_amount": 4
    }
}
```


Payload field	Data type	Description	Required	Stored
casino_last_bet_amount Bet Amount	monetary (numeric)	Bet amount in the user currency (total of bonus and real) `payload: { casino_last_bet_amount: 5 }`	YES	Dynamic

Payload field	Data type	Description	Required	Stored

casino_last_bet_game_name
Bet Game Name
	enumeration	Name of the game, e.g. "Super Slot"
		`payload: {`
		`  casino_last_bet_game_name: 'Super Slot'`
		`}`
			YES	Dynamic

casino_last_bet_id
Bet id
	string	Unique ID of the bet
		`payload: {`
		`  casino_last_bet_id: 'betid-123552355'`
		`}`
			YES	Dynamic

casino_last_win_amount
Last Win Amount
	monetary (numeric)	Win amount in the player currency (total of real and bonus money). In case of lost should be 0.
		`payload: {`
		`  casino_last_win_amount: 10`
		`}`
			YES	Dynamic

acc_bonus_balance
Bonus balance
	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction
		`payload: {`
		`  acc_bonus_balance: 20.98`
		`}`
			-	6

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		`payload: {`
		`  acc_gc_balance: 50.25`
		`}`
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		`payload: {`
		`  acc_gc_bonus_balance: 50.25`
		`}`
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		`payload: {`
		`  acc_gc_real_balance: 50.25`
		`}`
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	`payload: {`
		`  acc_land_balance_bonus: 50.25`
		`}`
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	`payload: {`
		`  acc_land_balance_real: 50.25`
		`}`
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	`payload: {`
		`  acc_land_balance_total: 50.25`
		`}`
			-	1

Payload field	Data type	Description	Required	Stored
acc_last_active_currency Last active currency	enumeration	In case of multi-wallet accounts, should indicate currency of the current active wallet `payload: { acc_last_active_currency: 'USD' }` Possible values: USD	-	2
acc_real_balance Real balance	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction `payload: { acc_real_balance: 50.25 }`	-	10
acc_sc_balance Regular coins balance	numeric	Regular coins balance `payload: { acc_sc_balance: 50.25 }`	-	1
acc_sc_bonus_balance Regular coins bonus balance	numeric	Regular coins bonus balance `payload: { acc_sc_bonus_balance: 50.25 }`	-	1
acc_sc_real_balance Regular coins real balance	numeric	Regular coins real balance `payload: { acc_sc_real_balance: 50.25 }`	-	1
acc_total_balance Total balance	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction `payload: { acc_total_balance: 50.25 }`	-	1
casino_avg_bet_amount_real Average real money bet amount	monetary (numeric)	Average lifetime bet amount of user in real money `payload: { casino_avg_bet_amount_real: 10 }`	-	1
casino_game_ext_id Game external ID (from catalog)	string	ID of the Game on the operator side `payload: { casino_game_ext_id: 'Some value' }`	-	Dynamic
casino_game_provider_ext_id Game provider external ID (from catalog)	string	ID of the Game provider on the operator side `payload: { casino_game_provider_ext_id: 'Some value' }`	-	Dynamic

Payload field	Data type	Description	Required	Stored

casino_game_type_ext_id
Game type external ID (from catalog)
	string	ID of the Game provider on the operator side
		`payload: {`
		`  casino_game_type_ext_id: 'Some value'`
		`}`
			-	Dynamic

casino_gc_bet_amount
Casino bet amount (gold coins)
	numeric	Bet amount in the sweepstake gold coins
		`payload: {`
		`  casino_gc_bet_amount: 4`
		`}`
			-	Dynamic

casino_gc_win_amount
Casino win amount (gold coins)
	numeric	Win amount in the sweepstake gold coins
		`payload: {`
		`  casino_gc_win_amount: 4`
		`}`
			-	Dynamic

casino_is_free_bet
Is free bet
	boolean	Indicator if it was free bet
		`payload: {`
		`  casino_is_free_bet: true`
		`}`
			-	Dynamic

casino_last_bet_amount_bonus
Bet Amount (Bonus)
	monetary (numeric)	Bet amount in the user currency, bonus money.
		`payload: {`
		`  casino_last_bet_amount_bonus: 1`
		`}`
			-	Dynamic

casino_last_bet_amount_real
Bet Amount (Real)
	monetary (numeric)	Bet amount in the user currency, real money
		`payload: {`
		`  casino_last_bet_amount_real: 4`
		`}`
			-	Dynamic

casino_last_bet_game_provider
Bet Game Provider
	enumeration	Game provider, e.g. "Playtech"
		`payload: {`
		`  casino_last_bet_game_provider: 'Playtech'`
		`}`
			-	Dynamic

casino_last_bet_game_type
Bet Game Type
	enumeration	Game type, e.g. "Slots"
		`payload: {`
		`  casino_last_bet_game_type: 'Slots'`
		`}`
			-	Dynamic

casino_last_bet_game_type_arr
Bet Game Types
	enum array	Set of game types, e.g. "Slots" & "Super slots". Used when platform has multiple types assigned to the game
		`payload: {`
		`  casino_last_bet_game_type_arr: ["Value 1","Value 2"]`
		`}`
		Possible values: VALUE 1, VALUE 2
			-	Dynamic

casino_last_bet_game_vendor
Bet Game Vendor
	enumeration	Game vendor, e.g. "BTG"
		`payload: {`
		`  casino_last_bet_game_vendor: 'BTG'`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored
casino_last_bet_platform Bet platform	enumeration	The platform type from which the bet was done, e.g. WEB, MOBILE, APP etc. Important: property supports up to 10 unique values. `payload: { casino_last_bet_platform: 'WEB' }` Possible values: WEB	-	Dynamic
casino_last_bet_profit Last bet profit (Real money win minus Real money bet)	monetary (numeric)	Player profit amount in the player currency (real money part). `payload: { casino_last_bet_profit: 8 }`	-	Dynamic
casino_last_bet_profit_total Last bet profit (Total win minus Total bet)	monetary (numeric)	Player profit amount in the player currency (real and bonus money part). `payload: { casino_last_bet_profit_total: 8 }`	-	Dynamic
casino_last_bet_status Bet Status	enumeration	Optional status of the bet, for example can be used to indicate that win was marked as "rollback" `payload: { casino_last_bet_status: 'Rollback' }` Possible values: ROLLBACK, WIN	-	Dynamic
casino_last_win_amount_bonus Win Amount (Bonus)	monetary (numeric)	Win amount in the player currency (bonus money part). In case of lost should be 0. `payload: { casino_last_win_amount_bonus: 8 }`	-	Dynamic
casino_last_win_amount_real Win Amount (Real)	monetary (numeric)	Win amount in the player currency (real money part). In case of lost should be 0. `payload: { casino_last_win_amount_real: 8 }`	-	Dynamic
casino_sc_bet_amount Casino bet amount (regular coins)	numeric	Bet amount in the sweepstake coins (regular coins) `payload: { casino_sc_bet_amount: 4 }`	-	Dynamic
casino_sc_win_amount Casino win amount (regular coins)	numeric	Win amount in the sweepstake regular coins `payload: { casino_sc_win_amount: 4 }`	-	Dynamic

BET-WIN

Event type: casino_bet_win

Example

TEST


```
{
    "eid": "e89950ff-155e-4468-ab0e-6e7159945e77",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "casino_bet_win",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_gc_balance": 50.25,
        "acc_gc_bonus_balance": 50.25,
        "acc_gc_real_balance": 50.25,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_active_currency": "USD",
        "acc_real_balance": 50.25,
        "acc_sc_balance": 50.25,
        "acc_sc_bonus_balance": 50.25,
        "acc_sc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "casino_avg_bet_amount_real": 10,
        "casino_game_ext_id": "Some value",
        "casino_game_provider_ext_id": "Some value",
        "casino_game_type_ext_id": "Some value",
        "casino_gc_bet_amount": 4,
        "casino_gc_win_amount": 4,
        "casino_is_free_bet": "true",
        "casino_last_bet_amount": 5,
        "casino_last_bet_amount_bonus": 1,
        "casino_last_bet_amount_real": 4,
        "casino_last_bet_dt": 1757602862849,
        "casino_last_bet_game_name": "Super Slot",
        "casino_last_bet_game_provider": "Playtech",
        "casino_last_bet_game_type": "Slots",
        "casino_last_bet_game_type_arr": [
            "Value 1",
            "Value 2"
        ],
        "casino_last_bet_game_vendor": "BTG",
        "casino_last_bet_id": "betid-123552355",
        "casino_last_bet_platform": "WEB",
        "casino_last_bet_profit": 8,
        "casino_last_bet_profit_total": 8,
        "casino_last_bet_real_dt": 1757602862849,
        "casino_last_bet_status": "Rollback",
        "casino_last_session_avg_bet_amount": 5.51,
        "casino_last_session_bet_count": 5,
        "casino_last_win_amount": 10,
        "casino_last_win_amount_bonus": 8,
        "casino_last_win_amount_real": 8,
        "casino_sc_bet_amount": 4,
        "casino_sc_win_amount": 4
    }
}
```


Payload field	Data type	Description	Required	Stored
casino_last_bet_amount Bet Amount	monetary (numeric)	Bet amount in the user currency (total of bonus and real) `payload: { casino_last_bet_amount: 5 }`	YES	Dynamic

Payload field	Data type	Description	Required	Stored

casino_last_bet_dt
Last Bet Date
	timestamp	Date/time of the bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  casino_last_bet_dt: 1757602862923`
		`}`
			YES	8

casino_last_bet_game_name
Bet Game Name
	enumeration	Name of the game, e.g. "Super Slot"
		`payload: {`
		`  casino_last_bet_game_name: 'Super Slot'`
		`}`
			YES	Dynamic

casino_last_bet_id
Bet id
	string	Unique ID of the bet
		`payload: {`
		`  casino_last_bet_id: 'betid-123552355'`
		`}`
			YES	Dynamic

casino_last_win_amount
Last Win Amount
	monetary (numeric)	Win amount in the player currency (total of real and bonus money). In case of lost should be 0.
		`payload: {`
		`  casino_last_win_amount: 10`
		`}`
			YES	Dynamic

acc_bonus_balance
Bonus balance
	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction
		`payload: {`
		`  acc_bonus_balance: 20.98`
		`}`
			-	6

acc_gc_balance
Gold coins balance
	numeric	Gold coins balance
		`payload: {`
		`  acc_gc_balance: 50.25`
		`}`
			-	1

acc_gc_bonus_balance
Gold coins bonus balance
	numeric	Gold coins bonus balance
		`payload: {`
		`  acc_gc_bonus_balance: 50.25`
		`}`
			-	1

acc_gc_real_balance
Gold coins balance (real)
	numeric	Gold coins real balance
		`payload: {`
		`  acc_gc_real_balance: 50.25`
		`}`
			-	1

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	`payload: {`
		`  acc_land_balance_bonus: 50.25`
		`}`
			-	1

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	`payload: {`
		`  acc_land_balance_real: 50.25`
		`}`
			-	1

Payload field	Data type	Description	Required	Stored

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_last_active_currency
Last active currency
	enumeration	In case of multi-wallet accounts, should indicate currency of the current active wallet
		payload: {
		acc_last_active_currency: 'USD'
		}
		Possible values: USD
			-	2

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_sc_balance
Regular coins balance
	numeric	Regular coins balance
		payload: {
		acc_sc_balance: 50.25
		}
			-	1

acc_sc_bonus_balance
Regular coins bonus balance
	numeric	Regular coins bonus balance
		payload: {
		acc_sc_bonus_balance: 50.25
		}
			-	1

acc_sc_real_balance
Regular coins real balance
	numeric	Regular coins real balance
		payload: {
		acc_sc_real_balance: 50.25
		}
			-	1

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		payload: {
		acc_total_balance: 50.25
		}
			-	1

casino_avg_bet_amount_real
Average real money bet amount
	monetary (numeric)	Average lifetime bet amount of user in real money
		payload: {
		casino_avg_bet_amount_real: 10
		}
			-	1

casino_game_ext_id
Game external ID (from catalog)
	string	ID of the Game on the operator side
		payload: {
		casino_game_ext_id: 'Some value'
		}
			-	Dynamic

Payload field	Data type	Description	Required	Stored

casino_game_provider_ext_id
Game provider external ID (from catalog)
	string	ID of the Game provider on the operator side
		`payload: {`
		`  casino_game_provider_ext_id: 'Some value'`
		`}`
			-	Dynamic

casino_game_type_ext_id
Game type external ID (from catalog)
	string	ID of the Game provider on the operator side
		`payload: {`
		`  casino_game_type_ext_id: 'Some value'`
		`}`
			-	Dynamic

casino_gc_bet_amount
Casino bet amount (gold coins)
	numeric	Bet amount in the sweepstake gold coins
		`payload: {`
		`  casino_gc_bet_amount: 4`
		`}`
			-	Dynamic

casino_gc_win_amount
Casino win amount (gold coins)
	numeric	Win amount in the sweepstake gold coins
		`payload: {`
		`  casino_gc_win_amount: 4`
		`}`
			-	Dynamic

casino_is_free_bet
Is free bet
	boolean	Indicator if it was free bet
		`payload: {`
		`  casino_is_free_bet: true`
		`}`
			-	Dynamic

casino_last_bet_amount_bonus
Bet Amount (Bonus)
	monetary (numeric)	Bet amount in the user currency, bonus money.
		`payload: {`
		`  casino_last_bet_amount_bonus: 1`
		`}`
			-	Dynamic

casino_last_bet_amount_real
Bet Amount (Real)
	monetary (numeric)	Bet amount in the user currency, real money
		`payload: {`
		`  casino_last_bet_amount_real: 4`
		`}`
			-	Dynamic

casino_last_bet_game_provider
Bet Game Provider
	enumeration	Game provider, e.g. "Playtech"
		`payload: {`
		`  casino_last_bet_game_provider: 'Playtech'`
		`}`
			-	Dynamic

casino_last_bet_game_type
Bet Game Type
	enumeration	Game type, e.g. "Slots"
		`payload: {`
		`  casino_last_bet_game_type: 'Slots'`
		`}`
			-	Dynamic

casino_last_bet_game_type_arr
Bet Game Types
	enum array	Set of game types, e.g. "Slots" & "Super slots". Used when platform has multiple types assigned to the game
		`payload: {`
		`  casino_last_bet_game_type_arr: ["Value 1","Value 2"]`
		`}`
		Possible values: VALUE 1, VALUE 2
			-	Dynamic

Payload field	Data type	Description	Required	Stored

casino_last_bet_game_vendor
Bet Game Vendor
	enumeration	Game vendor, e.g. "BTG"
		`payload: {`
		`  casino_last_bet_game_vendor: 'BTG'`
		`}`
			-	Dynamic

casino_last_bet_platform
Bet platform
	enumeration	The platform type from which the bet was done, e.g. WEB, MOBILE, APP etc.
		Important: property supports up to 10 unique values.
		`payload: {`
		`  casino_last_bet_platform: 'WEB'`
		`}`
		Possible values: WEB
			-	Dynamic

casino_last_bet_profit
Last bet profit (Real money win minus Real money bet)
	monetary (numeric)	Player profit amount in the player currency (real money part).
		`payload: {`
		`  casino_last_bet_profit: 8`
		`}`
			-	Dynamic

casino_last_bet_profit_total
Last bet profit (Total win minus Total bet)
	monetary (numeric)	Player profit amount in the player currency (real and bonus money part).
		`payload: {`
		`  casino_last_bet_profit_total: 8`
		`}`
			-	Dynamic

casino_last_bet_real_dt
Last Real Money Bet Date
	timestamp	Date/time of the bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  casino_last_bet_real_dt: 1757602862927`
		`}`
			-	

casino_last_bet_status
Bet Status
	enumeration	Optional status of the bet, for example can be used to indicate that win was marked as "rollback"
		`payload: {`
		`  casino_last_bet_status: 'Rollback'`
		`}`
		Possible values: ROLLBACK, WIN
			-	Dynamic

casino_last_session_avg_bet_amount
Last Session Bet Count
	monetary (numeric)	The bet count in the last casino session
		`payload: {`
		`  casino_last_session_avg_bet_amount: 5.51`
		`}`
			-	Dynamic

casino_last_session_bet_count
Last Session Bet Count
	numeric	The bet count in the last casino session
		`payload: {`
		`  casino_last_session_bet_count: 5`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored

casino_last_win_amount_bonus
Win Amount (Bonus)
	monetary (numeric)	Win amount in the player currency (bonus money part). In case of lost should be 0.
		`payload: {`
		`  casino_last_win_amount_bonus: 8`
		`}`
			-	Dynamic

casino_last_win_amount_real
Win Amount (Real)
	monetary (numeric)	Win amount in the player currency (real money part). In case of lost should be 0.
		`payload: {`
		`  casino_last_win_amount_real: 8`
		`}`
			-	Dynamic

casino_sc_bet_amount
Casino bet amount (regular coins)
	numeric	Bet amount in the sweepstake coins (regular coins)
		`payload: {`
		`  casino_sc_bet_amount: 4`
		`}`
			-	Dynamic

casino_sc_win_amount
Casino win amount (regular coins)
	numeric	Win amount in the sweepstake regular coins
		`payload: {`
		`  casino_sc_win_amount: 4`
		`}`
			-	Dynamic

SPORT EVENTS

There are 4 important events in the Sports domain - Bet, Selection open, Selection settled, and Bet settled. It's important to have the same "Bet ID" passed in all 4 events in order Smartico system to be able to connect these events in one logical sequence.

BET SETTLED

Event type: sport_bet_settled

Example TEST


```
{
    "eid": "5527c990-c41f-49ae-bf32-ebbc95c77f65",
    "event_date": 1757602862849,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "sport_bet_settled",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_active_currency": "USD",
        "acc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "sport_bet_original_date": 1757602862849,
        "sport_correct_score": "2:0",
        "sport_gc_bet_amount": 4,
        "sport_gc_win_amount": 4,
        "sport_last_bet_amount": 1.25,
        "sport_last_bet_amount_bonus": 0.25,
        "sport_last_bet_amount_real": 1,
        "sport_last_bet_competitors": [
            "ManU",
            "PSG"
        ],
        "sport_last_bet_id": "betid-123552355",
        "sport_last_bet_leagues": [
            "NFL",
            "NBA"
        ],
        "sport_last_bet_markets": [
            "1x2",
            "2x1"
        ],
        "sport_last_bet_odds": 1.25,
        "sport_last_bet_platform": "WEB",
        "sport_last_bet_profit": 8,
        "sport_last_bet_profit_total": 8,
        "sport_last_bet_provider": "SportRadar",
        "sport_last_bet_selection_count": 5,
        "sport_last_bet_sports": [
            "Soccer",
            "Tennis"
        ],
        "sport_last_bet_timing": "LIVE",
        "sport_last_bet_type": "SINGLE",
        "sport_last_bet_virtual": "true",
        "sport_last_bet_win_amount": 3,
        "sport_last_bet_win_amount_bonus": 2,
        "sport_last_bet_win_amount_real": 2,
        "sport_last_min_selection_odds": 1.25,
        "sport_last_settlement_status": "WON",
        "sport_match_id": "123456",
        "sport_original_bet_date": 1757602862850,
        "sport_sc_bet_amount": 4,
        "sport_sc_win_amount": 4,
        "sport_total_open_bets": 3
    }
}
```

Payload field	Data type	Description	Required	Stored
sport_last_bet_amount Bet Amount	monetary (numeric)	The bet amount in the user currency, the total amount that includes real and bonus money. This amount should include ALL selections in the bet and value should be passed in all 4 events - bet open, selection open, selection settled, bet settled `payload: { sport_last_bet_amount: 1.25 }`	YES	Dynamic
sport_last_bet_competitors Last Bet Competitors	enum array	List of Competitors within the bet selections `payload: { sport_last_bet_competitors: ["ManU","PSG"] }`	YES	Dynamic
sport_last_bet_id Bet Id	string	Unique ID of the sport bet. Important: same bet ID should be passed in all 4 sport related events - Bet open, Selection open, Selection settled, Bet settled `payload: { sport_last_bet_id: 'betid-123552355' }`	YES	Dynamic
sport_last_bet_leagues Last Bet Leagues	enum array	List of Leagues within the bet selections `payload: { sport_last_bet_leagues: ["NFL","NBA"] }`	YES	Dynamic
sport_last_bet_odds Bet Odds	numeric	Sport BET odds. `payload: { sport_last_bet_odds: 1.25 }`	YES	Dynamic
sport_last_bet_sports Last Bet Sports	enum array	List of sport types within the bet selections `payload: { sport_last_bet_sports: ["Soccer","Tennis"] }`	YES	Dynamic
sport_last_bet_type Bet Type	enumeration	Type of sport bet, can be named as COMBO, SINGLE or SYSTEM. Or up to integrated system naming convention. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled `payload: { sport_last_bet_type: 'SINGLE' }` Possible values: MULTIPLE, SINGLE	YES	Dynamic
sport_last_bet_win_amount Win Amount	monetary (numeric)	Total win amount in player currency. Real & Bonus money `payload: { sport_last_bet_win_amount: 3 }`	YES	Dynamic
sport_last_settlement_status Bet Settlement Status	enumeration	Final status of the sport bet on settlement. Can be one of "WON" | "LOST" | "VOID" | "CASHOUT" | "CANCELED" | "PARTIAL_CASHOUT" `payload: { sport_last_settlement_status: 'WON' }`	YES	Dynamic

Payload field	Data type	Description	Required	Stored
acc_bonus_balance Bonus balance	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction `payload: { acc_bonus_balance: 20.98 }`	-	6
acc_land_balance_bonus Bonus balance, landbase	monetary (numeric)	`payload: { acc_land_balance_bonus: 50.25 }`	-	1
acc_land_balance_real Real balance, landbase	monetary (numeric)	`payload: { acc_land_balance_real: 50.25 }`	-	1
acc_land_balance_total Total balance, landbase	monetary (numeric)	`payload: { acc_land_balance_total: 50.25 }`	-	1
acc_last_active_currency Last active currency	enumeration	In case of multi-wallet accounts, should indicate currency of the current active wallet `payload: { acc_last_active_currency: 'USD' }` Possible values: USD	-	2
acc_real_balance Real balance	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction `payload: { acc_real_balance: 50.25 }`	-	10
acc_total_balance Total balance	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction `payload: { acc_total_balance: 50.25 }`	-	1
sport_bet_original_date Original Bet Date	timestamp	Date of the original sport bet The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { sport_bet_original_date: 1757602862930 }`	-	Dynamic
sport_correct_score Correct score	string	The exact score of resolved bet. Note that it make sense for the "sport_bet_selection_settled" event and could make sense for the "sport_bet_settled" event when there is only one selection in the bet `payload: { sport_correct_score: '2:0' }`	-	Dynamic

Payload field	Data type	Description	Required	Stored

sport_gc_bet_amount
Sport bet amount (gold coins)
	numeric	Bet amount in the sweepstake gold coins, sport bets
		`payload: {`
		`  sport_gc_bet_amount: 4`
		`}`
			-	Dynamic

sport_gc_win_amount
Sport win amount (gold coins)
	numeric	Win amount in the sweepstake gold coins, sport bets
		`payload: {`
		`  sport_gc_win_amount: 4`
		`}`
			-	Dynamic

sport_last_bet_amount_bonus
Bet Amount (Bonus)
	monetary (numeric)	Bet amount in the user currency, bonus money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		`payload: {`
		`  sport_last_bet_amount_bonus: 0.25`
		`}`
			-	Dynamic

sport_last_bet_amount_real
Bet Amount (Real)
	monetary (numeric)	Bet amount in the user currency, real money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		`payload: {`
		`  sport_last_bet_amount_real: 1`
		`}`
			-	Dynamic

sport_last_bet_markets
Last Bet Markets
	enum array	List of Markets within the bet selections
		`payload: {`
		`  sport_last_bet_markets: ["1x2","2x1"]`
		`}`
			-	Dynamic

sport_last_bet_platform
Bet platform
	enumeration	The platform type from which the bet was done, e.g. WEB, MOBILE, APP etc.
		Important: property supports up to 20 unique values.
		`payload: {`
		`  sport_last_bet_platform: 'WEB'`
		`}`
			-	Dynamic

sport_last_bet_profit
Last bet profit (Real money win minus Real money bet)
	monetary (numeric)	Player profit amount in the player currency (real money part).
		`payload: {`
		`  sport_last_bet_profit: 8`
		`}`
			-	Dynamic

sport_last_bet_profit_total
Last bet profit (Total win minus Total bet)
	monetary (numeric)	Player profit amount in the player currency (real and bonus money part).
		`payload: {`
		`  sport_last_bet_profit_total: 8`
		`}`
			-	Dynamic

sport_last_bet_provider
Bet platform
	enumeration	The name of sport provider
		Important: property supports up to 50 unique values.
		`payload: {`
		`  sport_last_bet_provider: 'SportRadar'`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored

sport_last_bet_selection_count
Last Bet Selection Count
	numeric	Selections Count in the Last Sport Bet
		`payload: {`
		`  sport_last_bet_selection_count: 5`
		`}`
			-	Dynamic

sport_last_bet_timing
Bet Timing
	enumeration	LIVE or PREMATCH
		`payload: {`
		`  sport_last_bet_timing: 'LIVE'`
		`}`
		Possible values: PREMATCH
			-	Dynamic

sport_last_bet_virtual
Bet is Virtual
	boolean	Is last bet as on virtual sport
		`payload: {`
		`  sport_last_bet_virtual: true`
		`}`
			-	Dynamic

sport_last_bet_win_amount_bonus
Win Amount (Bonus)
	monetary (numeric)	Win amount, bonus money part in user currency
		`payload: {`
		`  sport_last_bet_win_amount_bonus: 2`
		`}`
			-	Dynamic

sport_last_bet_win_amount_real
Win Amount (Real)
	monetary (numeric)	Win amount, real money part in user currency
		`payload: {`
		`  sport_last_bet_win_amount_real: 2`
		`}`
			-	Dynamic

sport_last_min_selection_odds
Minimum odds in the selections of the Bet
	numeric	The minimum odds in the selections of the Bet
		`payload: {`
		`  sport_last_min_selection_odds: 1.25`
		`}`
			-	Dynamic

sport_match_id
Match ID
	string	ID of the sport match
		`payload: {`
		`  sport_match_id: '123456'`
		`}`
			-	Dynamic

sport_original_bet_date
Original Bet Date
	timestamp	Date of the original sport bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  sport_original_bet_date: 1757602862933`
		`}`
			-	Dynamic

sport_sc_bet_amount
Sport bet amount (regular coins)
	numeric	Bet amount in the sweepstake regular coins, sport bets
		`payload: {`
		`  sport_sc_bet_amount: 4`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored

sport_sc_win_amount
Sport win amount (regular coins)
	numeric	Win amount in the sweepstake regular coins, sport bets
		`payload: {`
		`  sport_sc_win_amount: 4`
		`}`
			-	Dynamic

sport_total_open_bets
Total / Lifetime bets count
	numeric	Number of sports betsa user has ever opened
		`payload: {`
		`  sport_total_open_bets: 3`
		`}`
			-	

BET OPEN

Event type: sport_bet_open

Example

TEST


```
{
    "eid": "c2edc833-ca9f-4cce-9af5-6dac8caada26",
    "event_date": 1757602862850,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "sport_bet_open",
    "payload": {
        "acc_bonus_balance": 20.98,
        "acc_land_balance_bonus": 50.25,
        "acc_land_balance_real": 50.25,
        "acc_land_balance_total": 50.25,
        "acc_last_active_currency": "USD",
        "acc_real_balance": 50.25,
        "acc_total_balance": 50.25,
        "sport_first_bet_date": 1757602862850,
        "sport_gc_bet_amount": 4,
        "sport_last_bet_amount": 1.25,
        "sport_last_bet_amount_bonus": 0.25,
        "sport_last_bet_amount_real": 1,
        "sport_last_bet_competitors": [
            "ManU",
            "PSG"
        ],
        "sport_last_bet_date": 1757602862850,
        "sport_last_bet_id": "betid-123552355",
        "sport_last_bet_leagues": [
            "NFL",
            "NBA"
        ],
        "sport_last_bet_markets": [
            "1x2",
            "2x1"
        ],
        "sport_last_bet_odds": 1.25,
        "sport_last_bet_platform": "WEB",
        "sport_last_bet_provider": "SportRadar",
        "sport_last_bet_real_dt": 1757602862850,
        "sport_last_bet_selection_count": 5,
        "sport_last_bet_sports": [
            "Soccer",
            "Tennis"
        ],
        "sport_last_bet_timing": "LIVE",
        "sport_last_bet_type": "SINGLE",
        "sport_last_bet_virtual": "true",
        "sport_last_min_selection_odds": 1.25,
        "sport_match_id": "123456",
        "sport_sc_bet_amount": 4,
        "sport_total_open_bets": 3
    }
}
```


Payload field	Data type	Description	Required	Stored
sport_last_bet_amount Bet Amount	monetary (numeric)	The bet amount in the user currency, the total amount that includes real and bonus money. This amount should include ALL selections in the bet and value should be passed in all 4 events - bet open, selection open, selection settled, bet settled `payload: { sport_last_bet_amount: 1.25 }`	YES	Dynamic

Payload field	Data type	Description	Required	Stored

sport_last_bet_competitors
Last Bet Competitors
	enum array	List of Competitors within the bet selections
		`payload: {`
		`  sport_last_bet_competitors: ["ManU","PSG"]`
		`}`
			YES	Dynamic

sport_last_bet_date
Last Bet Date
	timestamp	Date of the sport bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  sport_last_bet_date: 1757602862934`
		`}`
			YES	3

sport_last_bet_id
Bet Id
	string	Unique ID of the sport bet. Important: same bet ID should be passed in all 4 sport related events - Bet open, Selection open, Selection settled, Bet settled
		`payload: {`
		`  sport_last_bet_id: 'betid-123552355'`
		`}`
			YES	Dynamic

sport_last_bet_leagues
Last Bet Leagues
	enum array	List of Leagues within the bet selections
		`payload: {`
		`  sport_last_bet_leagues: ["NFL","NBA"]`
		`}`
			YES	Dynamic

sport_last_bet_odds
Bet Odds
	numeric	Sport BET odds.
		`payload: {`
		`  sport_last_bet_odds: 1.25`
		`}`
			YES	Dynamic

sport_last_bet_sports
Last Bet Sports
	enum array	List of sport types within the bet selections
		`payload: {`
		`  sport_last_bet_sports: ["Soccer","Tennis"]`
		`}`
			YES	Dynamic

sport_last_bet_type
Bet Type
	enumeration	Type of sport bet, can be named as COMBO, SINGLE or SYSTEM. Or up to integrated system naming convention. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		`payload: {`
		`  sport_last_bet_type: 'SINGLE'`
		`}`
		Possible values: MULTIPLE, SINGLE
			YES	Dynamic

acc_bonus_balance
Bonus balance
	monetary (numeric)	Bonus money balance in the user currency. Recommended to be updated after every monetary operation that is changing bonus balance with the state of balance after transaction
		`payload: {`
		`  acc_bonus_balance: 20.98`
		`}`
			-	6

acc_land_balance_bonus
Bonus balance, landbase
	monetary (numeric)	`payload: {`
		`  acc_land_balance_bonus: 50.25`
		`}`
			-	1

Payload field	Data type	Description	Required	Stored

acc_land_balance_real
Real balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_real: 50.25
		}
			-	1

acc_land_balance_total
Total balance, landbase
	monetary (numeric)	payload: {
		acc_land_balance_total: 50.25
		}
			-	1

acc_last_active_currency
Last active currency
	enumeration	In case of multi-wallet accounts, should indicate currency of the current active wallet
		payload: {
		acc_last_active_currency: 'USD'
		}
		Possible values: USD
			-	2

acc_real_balance
Real balance
	monetary (numeric)	Real money balance in the user currency. Recommended to be updated after every monetary operation that is changing real balance with the state of balance after the transaction
		payload: {
		acc_real_balance: 50.25
		}
			-	10

acc_total_balance
Total balance
	monetary (numeric)	Real + bonuse money balance in the user currency. Recommended to be updated after every monetary operation that is changing real/bonus balances with the state of balance after the transaction
		payload: {
		acc_total_balance: 50.25
		}
			-	1

sport_first_bet_date
First Bet Date
	timestamp	Date of the first sport bet
		The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past.
		payload: {
		sport_first_bet_date: 1757602862935
		}
			-	

sport_gc_bet_amount
Sport bet amount (gold coins)
	numeric	Bet amount in the sweepstake gold coins, sport bets
		payload: {
		sport_gc_bet_amount: 4
		}
			-	Dynamic

sport_last_bet_amount_bonus
Bet Amount (Bonus)
	monetary (numeric)	Bet amount in the user currency, bonus money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		payload: {
		sport_last_bet_amount_bonus: 0.25
		}
			-	Dynamic

sport_last_bet_amount_real
Bet Amount (Real)
	monetary (numeric)	Bet amount in the user currency, real money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		payload: {
		sport_last_bet_amount_real: 1
		}
			-	Dynamic

Payload field	Data type	Description	Required	Stored
sport_last_bet_markets Last Bet Markets	enum array	List of Markets within the bet selections `payload: { sport_last_bet_markets: ["1x2","2x1"] }`	-	Dynamic
sport_last_bet_platform Bet platform	enumeration	The platform type from which the bet was done, e.g. WEB, MOBILE, APP etc. Important: property supports up to 20 unique values. `payload: { sport_last_bet_platform: 'WEB' }`	-	Dynamic
sport_last_bet_provider Bet platform	enumeration	The name of sport provider Important: property supports up to 50 unique values. `payload: { sport_last_bet_provider: 'SportRadar' }`	-	Dynamic
sport_last_bet_real_dt Last Real Money Bet Date	timestamp	Date/time of the bet The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past. `payload: { sport_last_bet_real_dt: 1757602862936 }`	-	
sport_last_bet_selection_count Last Bet Selection Count	numeric	Selections Count in the Last Sport Bet `payload: { sport_last_bet_selection_count: 5 }`	-	Dynamic
sport_last_bet_timing Bet Timing	enumeration	LIVE or PREMATCH `payload: { sport_last_bet_timing: 'LIVE' }` Possible values: PREMATCH	-	Dynamic
sport_last_bet_virtual Bet is Virtual	boolean	Is last bet as on virtual sport `payload: { sport_last_bet_virtual: true }`	-	Dynamic
sport_last_min_selection_odds Minimum odds in the selections of the Bet	numeric	The minimum odds in the selections of the Bet `payload: { sport_last_min_selection_odds: 1.25 }`	-	Dynamic
sport_match_id Match ID	string	ID of the sport match `payload: { sport_match_id: '123456' }`	-	Dynamic

Payload field	Data type	Description	Required	Stored

sport_sc_bet_amount
Sport bet amount (regular coins)
	numeric	Bet amount in the sweepstake regular coins, sport bets
		`payload: {`
		`  sport_sc_bet_amount: 4`
		`}`
			-	Dynamic

sport_total_open_bets
Total / Lifetime bets count
	numeric	Number of sports betsa user has ever opened
		`payload: {`
		`  sport_total_open_bets: 3`
		`}`
			-	

SELECTION SETTLED

Event type: sport_bet_selection_settled

Example TEST


```
{
    "eid": "af05b830-b7c0-483a-95bd-06015250f7fe",
    "event_date": 1757602862850,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "sport_bet_selection_settled",
    "payload": {
        "sport_bet_original_date": 1757602862850,
        "sport_correct_score": "2:0",
        "sport_gc_bet_amount": 4,
        "sport_gc_win_amount": 4,
        "sport_last_bet_amount": 1.25,
        "sport_last_bet_amount_bonus": 0.25,
        "sport_last_bet_amount_real": 1,
        "sport_last_bet_id": "betid-123552355",
        "sport_last_bet_platform": "WEB",
        "sport_last_bet_provider": "SportRadar",
        "sport_last_bet_selection_count": 5,
        "sport_last_bet_type": "SINGLE",
        "sport_last_min_selection_odds": 1.25,
        "sport_last_selection_away_team": "Some value",
        "sport_last_selection_bet_amount": 1.25,
        "sport_last_selection_competitors": [
            "Dynamo Kyiv",
            "AC Milan"
        ],
        "sport_last_selection_home_team": "Some value",
        "sport_last_selection_id": "betid-123552355",
        "sport_last_selection_is_live": "true",
        "sport_last_selection_league": "UEFA Champions League",
        "sport_last_selection_market": "Both teams to score",
        "sport_last_selection_odds": 1.25,
        "sport_last_selection_outcome": "3:5",
        "sport_last_selection_settlement_status": "WON",
        "sport_last_selection_sport_type": "Football",
        "sport_last_selection_virtual": "true",
        "sport_last_selection_win_amount": 1.25,
        "sport_match_id": "123456",
        "sport_original_bet_date": 1757602862850,
        "sport_sc_bet_amount": 4,
        "sport_sc_win_amount": 4
    }
}
```


Payload field	Data type	Description	Required	Stored
sport_last_bet_amount Bet Amount	monetary (numeric)	The bet amount in the user currency, the total amount that includes real and bonus money. This amount should include ALL selections in the bet and value should be passed in all 4 events - bet open, selection open, selection settled, bet settled `payload: { sport_last_bet_amount: 1.25 }`	YES	Dynamic
sport_last_bet_id Bet Id	string	Unique ID of the sport bet. Important: same bet ID should be passed in all 4 sport related events - Bet open, Selection open, Selection settled, Bet settled `payload: { sport_last_bet_id: 'betid-123552355' }`	YES	Dynamic

Payload field	Data type	Description	Required	Stored

sport_last_bet_type
Bet Type
	enumeration	Type of sport bet, can be named as COMBO, SINGLE or SYSTEM. Or up to integrated system naming convention. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		`payload: {`
		`  sport_last_bet_type: 'SINGLE'`
		`}`
		Possible values: MULTIPLE, SINGLE
			YES	Dynamic

sport_last_selection_competitors
Selection Competitors
	enum array	List of competitors
		`payload: {`
		`  sport_last_selection_competitors: ["Dynamo Kyiv","AC Milan"]`
		`}`
			YES	Dynamic

sport_last_selection_id
Last selection id
	string	Unique ID of the sport selection.
		`payload: {`
		`  sport_last_selection_id: 'betid-123552355'`
		`}`
			YES	Dynamic

sport_last_selection_is_live
Selection is Live
	boolean	"Is live" indicator for the selection. Should be also passed in the 'selection settlement' with original value as it was in 'selection open'
		`payload: {`
		`  sport_last_selection_is_live: true`
		`}`
			YES	Dynamic

sport_last_selection_league
Selection League Name
	enumeration	League name of specific selection
		`payload: {`
		`  sport_last_selection_league: 'UEFA Champions League'`
		`}`
			YES	Dynamic

sport_last_selection_market
Selection Market
	enumeration	Market of specific selection
		`payload: {`
		`  sport_last_selection_market: 'Both teams to score'`
		`}`
			YES	Dynamic

sport_last_selection_odds
Selection Odds
	numeric	Odds of the specific selection in sport bet. The same value should be passed in all 2 events - selection open, selection settled
		`payload: {`
		`  sport_last_selection_odds: 1.25`
		`}`
			YES	Dynamic

sport_last_selection_settlement_status
Selection Settlement Status
	enumeration	The status of specific selection
		`payload: {`
		`  sport_last_selection_settlement_status: 'WON'`
		`}`
			YES	Dynamic

sport_last_selection_sport_type
Selection Sport Type
	enumeration	Sport type of specific selection
		`payload: {`
		`  sport_last_selection_sport_type: 'Football'`
		`}`
			YES	Dynamic

Payload field	Data type	Description	Required	Stored

sport_bet_original_date
Original Bet Date
	timestamp	Date of the original sport bet
		The passed value should be a timestamp in UTC with milliseconds precision. The date value should be in the past.
		`payload: {`
		`  sport_bet_original_date: 1757602862939`
		`}`
			-	Dynamic

sport_correct_score
Correct score
	string	The exact score of resolved bet. Note that it make sense for the "sport_bet_selection_settled" event and could make sense for the "sport_bet_settled" event when there is only one selection in the bet
		`payload: {`
		`  sport_correct_score: '2:0'`
		`}`
			-	Dynamic

sport_gc_bet_amount
Sport bet amount (gold coins)
	numeric	Bet amount in the sweepstake gold coins, sport bets
		`payload: {`
		`  sport_gc_bet_amount: 4`
		`}`
			-	Dynamic

sport_gc_win_amount
Sport win amount (gold coins)
	numeric	Win amount in the sweepstake gold coins, sport bets
		`payload: {`
		`  sport_gc_win_amount: 4`
		`}`
			-	Dynamic

sport_last_bet_amount_bonus
Bet Amount (Bonus)
	monetary (numeric)	Bet amount in the user currency, bonus money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		`payload: {`
		`  sport_last_bet_amount_bonus: 0.25`
		`}`
			-	Dynamic

sport_last_bet_amount_real
Bet Amount (Real)
	monetary (numeric)	Bet amount in the user currency, real money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		`payload: {`
		`  sport_last_bet_amount_real: 1`
		`}`
			-	Dynamic

sport_last_bet_platform
Bet platform
	enumeration	The platform type from which the bet was done, e.g. WEB, MOBILE, APP etc.
		Important: property supports up to 20 unique values.
		`payload: {`
		`  sport_last_bet_platform: 'WEB'`
		`}`
			-	Dynamic

sport_last_bet_provider
Bet platform
	enumeration	The name of sport provider
		Important: property supports up to 50 unique values.
		`payload: {`
		`  sport_last_bet_provider: 'SportRadar'`
		`}`
			-	Dynamic

sport_last_bet_selection_count
Last Bet Selection Count
	numeric	Selections Count in the Last Sport Bet
		`payload: {`
		`  sport_last_bet_selection_count: 5`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored

sport_last_min_selection_odds
Minimum odds in the selections of the Bet
	numeric	The minimum odds in the selections of the Bet
		`payload: {`
		`  sport_last_min_selection_odds: 1.25`
		`}`
			-	Dynamic

sport_last_selection_away_team
Selection Away Team
	enumeration	`payload: {`
		`  sport_last_selection_away_team: 'Some value'`
		`}`
			-	Dynamic

sport_last_selection_bet_amount
Selection Bet Amount
	numeric	Bet Amount of a specific selection in sport bet. The same value should be passed in all 2 events - selection open, selection settled
		`payload: {`
		`  sport_last_selection_bet_amount: 1.25`
		`}`
			-	Dynamic

sport_last_selection_home_team
Selection Home Team
	enumeration	`payload: {`
		`  sport_last_selection_home_team: 'Some value'`
		`}`
			-	Dynamic

sport_last_selection_outcome
Last selection outcome
	string	The match outcome, either picked by user on bet placement or the final at the moment of the settlement
		`payload: {`
		`  sport_last_selection_outcome: '3:5'`
		`}`
			-	Dynamic

sport_last_selection_virtual
Selection Sport is Virtual
	boolean	Is last bet as on virtual sport
		`payload: {`
		`  sport_last_selection_virtual: true`
		`}`
			-	Dynamic

sport_last_selection_win_amount
Selection Win Amount
	numeric	Win Amount of a specific selection in sport bet.
		`payload: {`
		`  sport_last_selection_win_amount: 1.25`
		`}`
			-	Dynamic

sport_match_id
Match ID
	string	ID of the sport match
		`payload: {`
		`  sport_match_id: '123456'`
		`}`
			-	Dynamic

sport_original_bet_date
Original Bet Date
	timestamp	Date of the original sport bet
		The passed value should be a timestamp in UTC with milliseconds precision.
		The date value should be in the past.
		`payload: {`
		`  sport_original_bet_date: 1757602862941`
		`}`
			-	Dynamic

sport_sc_bet_amount
Sport bet amount (regular coins)
	numeric	Bet amount in the sweepstake regular coins, sport bets
		`payload: {`
		`  sport_sc_bet_amount: 4`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored

sport_sc_win_amount
Sport win amount (regular coins)
	numeric	Win amount in the sweepstake regular coins, sport bets
		`payload: {`
		`  sport_sc_win_amount: 4`
		`}`
			-	Dynamic

SELECTION OPEN

Event type: sport_bet_selection_open

Example

TEST


```
{
    "eid": "c5ab171c-3fa8-4bc7-8cb6-487c2c3926b7",
    "event_date": 1757602862850,
    "ext_brand_id": null,
    "user_ext_id": "some_user_id_1",
    "event_type": "sport_bet_selection_open",
    "payload": {
        "sport_gc_bet_amount": 4,
        "sport_last_bet_amount": 1.25,
        "sport_last_bet_amount_bonus": 0.25,
        "sport_last_bet_amount_real": 1,
        "sport_last_bet_id": "betid-123552355",
        "sport_last_bet_platform": "WEB",
        "sport_last_bet_provider": "SportRadar",
        "sport_last_bet_selection_count": 5,
        "sport_last_bet_type": "SINGLE",
        "sport_last_min_selection_odds": 1.25,
        "sport_last_selection_away_team": "Some value",
        "sport_last_selection_bet_amount": 1.25,
        "sport_last_selection_competitors": [
            "Dynamo Kyiv",
            "AC Milan"
        ],
        "sport_last_selection_home_team": "Some value",
        "sport_last_selection_id": "betid-123552355",
        "sport_last_selection_is_live": "true",
        "sport_last_selection_league": "UEFA Champions League",
        "sport_last_selection_market": "Both teams to score",
        "sport_last_selection_odds": 1.25,
        "sport_last_selection_outcome": "3:5",
        "sport_last_selection_sport_type": "Football",
        "sport_last_selection_virtual": "true",
        "sport_last_selection_win_amount": 1.25,
        "sport_match_id": "123456",
        "sport_sc_bet_amount": 4
    }
}
```


Payload field	Data type	Description	Required	Stored
sport_last_bet_amount Bet Amount	monetary (numeric)	The bet amount in the user currency, the total amount that includes real and bonus money. This amount should include ALL selections in the bet and value should be passed in all 4 events - bet open, selection open, selection settled, bet settled `payload: { sport_last_bet_amount: 1.25 }`	YES	Dynamic
sport_last_bet_id Bet Id	string	Unique ID of the sport bet. Important: same bet ID should be passed in all 4 sport related events - Bet open, Selection open, Selection settled, Bet settled `payload: { sport_last_bet_id: 'betid-123552355' }`	YES	Dynamic

Payload field	Data type	Description	Required	Stored

sport_last_bet_type
Bet Type
	enumeration	Type of sport bet, can be named as COMBO, SINGLE or SYSTEM. Or up to integrated system naming convention. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled
		`payload: {`
		`  sport_last_bet_type: 'SINGLE'`
		`}`
		Possible values: MULTIPLE, SINGLE
			YES	Dynamic

sport_last_selection_competitors
Selection Competitors
	enum array	List of competitors
		`payload: {`
		`  sport_last_selection_competitors: ["Dynamo Kyiv","AC Milan"]`
		`}`
			YES	Dynamic

sport_last_selection_id
Last selection id
	string	Unique ID of the sport selection.
		`payload: {`
		`  sport_last_selection_id: 'betid-123552355'`
		`}`
			YES	Dynamic

sport_last_selection_is_live
Selection is Live
	boolean	"Is live" indicator for the selection. Should be also passed in the 'selection settlement' with original value as it was in 'selection open'
		`payload: {`
		`  sport_last_selection_is_live: true`
		`}`
			YES	Dynamic

sport_last_selection_league
Selection League Name
	enumeration	League name of specific selection
		`payload: {`
		`  sport_last_selection_league: 'UEFA Champions League'`
		`}`
			YES	Dynamic

sport_last_selection_market
Selection Market
	enumeration	Market of specific selection
		`payload: {`
		`  sport_last_selection_market: 'Both teams to score'`
		`}`
			YES	Dynamic

sport_last_selection_odds
Selection Odds
	numeric	Odds of the specific selection in sport bet. The same value should be passed in all 2 events - selection open, selection settled
		`payload: {`
		`  sport_last_selection_odds: 1.25`
		`}`
			YES	Dynamic

sport_last_selection_sport_type
Selection Sport Type
	enumeration	Sport type of specific selection
		`payload: {`
		`  sport_last_selection_sport_type: 'Football'`
		`}`
			YES	Dynamic

sport_gc_bet_amount
Sport bet amount (gold coins)
	numeric	Bet amount in the sweepstake gold coins, sport bets
		`payload: {`
		`  sport_gc_bet_amount: 4`
		`}`
			-	Dynamic

Payload field	Data type	Description	Required	Stored
sport_last_bet_amount_bonus Bet Amount (Bonus)	monetary (numeric)	Bet amount in the user currency, bonus money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled `payload: { sport_last_bet_amount_bonus: 0.25 }`	-	Dynamic
sport_last_bet_amount_real Bet Amount (Real)	monetary (numeric)	Bet amount in the user currency, real money. The same value should be passed in all 4 events - bet open, selection open, selection settled, bet settled `payload: { sport_last_bet_amount_real: 1 }`	-	Dynamic
sport_last_bet_platform Bet platform	enumeration	The platform type from which the bet was done, e.g. WEB, MOBILE, APP etc. Important: property supports up to 20 unique values. `payload: { sport_last_bet_platform: 'WEB' }`	-	Dynamic
sport_last_bet_provider Bet platform	enumeration	The name of sport provider Important: property supports up to 50 unique values. `payload: { sport_last_bet_provider: 'SportRadar' }`	-	Dynamic
sport_last_bet_selection_count Last Bet Selection Count	numeric	Selections Count in the Last Sport Bet `payload: { sport_last_bet_selection_count: 5 }`	-	Dynamic
sport_last_min_selection_odds Minimum odds in the selections of the Bet	numeric	The minimum odds in the selections of the Bet `payload: { sport_last_min_selection_odds: 1.25 }`	-	Dynamic
sport_last_selection_away_team Selection Away Team	enumeration	`payload: { sport_last_selection_away_team: 'Some value' }`	-	Dynamic
sport_last_selection_bet_amount Selection Bet Amount	numeric	Bet Amount of a specific selection in sport bet. The same value should be passed in all 2 events - selection open, selection settled `payload: { sport_last_selection_bet_amount: 1.25 }`	-	Dynamic
sport_last_selection_home_team Selection Home Team	enumeration	`payload: { sport_last_selection_home_team: 'Some value' }`	-	Dynamic
sport_last_selection_outcome Last selection outcome	string	The match outcome, either picked by user on bet placement or the final at the moment of the settlement `payload: { sport_last_selection_outcome: '3:5' }`	-	Dynamic

Payload field	Data type	Description	Required	Stored

sport_last_selection_virtual
Selection Sport is Virtual
	boolean	Is last bet as on virtual sport
		`payload: {`
		`  sport_last_selection_virtual: true`
		`}`
			-	Dynamic

sport_last_selection_win_amount
Selection Win Amount
	numeric	Win Amount of a specific selection in sport bet.
		`payload: {`
		`  sport_last_selection_win_amount: 1.25`
		`}`
			-	Dynamic

sport_match_id
Match ID
	string	ID of the sport match
		`payload: {`
		`  sport_match_id: '123456'`
		`}`
			-	Dynamic

sport_sc_bet_amount
Sport bet amount (regular coins)
	numeric	Bet amount in the sweepstake regular coins, sport bets
		`payload: {`
		`  sport_sc_bet_amount: 4`
		`}`
			-	Dynamic