Integration API specification

Latest update: Mar, 2025

API version 2.15.1

Version Control

Date	Version	Changes Description
01/20/2025	2.10	Add Separated Wallet
01/21/2025	2.11	Add GameName field in transaction callback and betting history
02/17/2025	2.12	Add Set User RTP API
02/20/2025	2.13	Add Get User RTP API, Reset Users RTP API
03/14/2025	2.14	Add Batch Transactions API
03/26/2025	2.15	Add Get Betting History V2 (by date) Deprecate Get Betting History (by date)
03/28/2025	2.15.1	- New Feature Add Get Betting Detail Page URL Add Rate Limit (1.4 Rate Limiting Policy). - Deprecated Feature Get Betting History (by date)

I. API Overview

Introducing GoldenGateX - The Cutting-Edge Live Casino, Slot, and Mini Game API Provider

In a groundbreaking move that is set to revolutionize the online gaming industry, GoldenGateX has officially launched its highly anticipated live casino, slot, and mini game API platform. This state-of-the-art solution promises to deliver an unparalleled gaming experience to players and operators alike.

Designed with innovation and excellence at its core, GoldenGateX offers a comprehensive suite of products that cater to the diverse needs of the modern gaming market. From immersive live casino experiences to thrilling slot games and engaging mini games, the platform has been meticulously crafted to provide a seamless and engaging journey for users.

1.1 Seamless Wallet API

Operator should provide the Seamless Wallet Integration API on their side. GoldenGateX will call the methods when players make a bet or get a win and their balance should be updated.

Method/Section	Description	Status
Balance	Returns the balance of a player.	Required
Transaction	Using this method a GoldenGateX system will notify Casino Operator about game transaction such as bet , win.	Required

1.2 Balance Transfer API

Operator can use this API for game opening and transferring funds to player’s balance in the GoldenGateX system.

Method/Section	Description	Status
CreateUser	This method allows registering a new player on GoldenGateX side.	Required
Balance	Returns the balance of a player.	Required
Deposit	This method transfers funds into player’s balance.	Required
Withdraw	This method transfers funds out of the player’s balance within GoldenGateX system	Required
WithdrawAll	This method transfers funds out of the player’s all balance within GoldenGateX system	Required

1.3 API Status

Request path: **GET** https://dcyqv8f2id.com /api/v2/status

This is a method to get information about the current state of the API. If everything is OK, you will receive an "OK" message. Otherwise, you'll see some error.

Example of HTTP request:

```
GET /api/v2/status HTTP/2
Host: xxx.com
```

Example of JSON response:

```
{
        "success": true,
        "message": "success",
        "errorCode": 0
}
```

1.4 API Rate Limiting Policy

To ensure fair usage and prevent abuse, our API enforces rate limits per endpoint. Each API endpoint has its own limit, which resets after a specific time period. If a user exceeds the limit, the API will return a 429 Too Many Requests error, and further requests must wait until the reset period.
Rate limits are communicated via response headers:

- **X-RateLimit-Limit** – The maximum number of requests allowed within the time window.
- **X-RateLimit-Remaining** – The number of requests remaining before hitting the limit.
- **X-RateLimit-Reset** – The Unix timestamp when the limit resets.

Rate Limits Per Endpoint

Endpoint	Limit	Time Window
POST /auth/createtoken	5	30 seconds
POST /betting/history/by-date-v2	1	1 second

II. Integration API

This is a simple API for Casino Operator to retrieve some data from GoldeGateX system and send some events to
GoldeGateX. API is an HTTP listener, which listens POST requests coming to an URL with the request mappings below.
All responses are in JSON format.

URL of the HTTP service will be provided by GoldenGateX for the production and test environments and looks
like:

```
https://dcyqv8f2id.com/api/v2
```

Integration simple HTTP Service is securely protected, hence please be sure:
- player's browser (end) must NOT be used as initiator of requests to the service;
- proper IPs are supplied to GoldenGateX for whitelisting

2.1 CreateToken

Using this method Casino Operator will get the authentication bearer token.
This API allows a maximum of **5 requests per 30 seconds**. Exceeding this limit may result in temporary throttling or request rejection.
Caution: Excessive requests for token generation may result in account blocking.
Request path: **POST** */auth/createtoken*

Request parameters

Name	Type	Description	Status
clientId	string	This is GoldenGateX agent clientId, you can find this in agent page.	Required
clientSecret	string	This is GoldenGateX agent clientSecret, you can find this in agent page.	Required

Response parameters

Name	Type	Description
token	string(250)	This is the Goldengatex token used for the request bearer token.
expiration	long	This is the expiration timestamp for this token.(seconds)

Example of HTTP request:

```
POST /api/v2/auth/createtoken HTTP/2
Host: xxx.com
content-type: application/json
accept: */*
Request Body
{
        "clientId": "testId",
        "clientSecret": "testSecret"
}
```

Example of JSON response:

```
{
        "token": "eyJhbGciOiJIUzUxMiI..........SlDhiXdB2y5Q",
        "expiration": 1716257131
}
```

2.2 Vendor List

Using this method Casino Operator will get the list of game providers available for the integration.
Request path: **GET** */vendors/list*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	List of Vendor	The list of vendors.(see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: Vendor

Name	Type	Description
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX
type	int	Game indicator of GoldenGateX 1: live casino 2: slot 3: mini-game
name	string	Vendor name provided by GoldenGateX
url	string	Vendor api endpoint

Example of HTTP request:

```
GET /api/v2/vendors/list HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
```

Example of JSON response:

```
{
        "success": true,
        "message": [
                {
                        "vendorCode": "slot-pragmatic",
                        "type": 2,
                        "name": "Pragmatic Slot",
                },
                {
                        "vendorCode": "mini-crash",
                        "type": 3,
                        "name": "Crash"
                },
                ...
                {
                        "vendorCode": "slot-habanero",
                        "type": 2,
                        "name": "Habanero"
                }
        ],
        "errorCode": 0
}
```

2.3 Game List

Using this method Casino Operator will get the list of casino games available for the integration.

Request path: **POST** */games/list*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.	Required
language	string	Language code. Ie. en, th,…	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	List of Game	The list of games.(see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: Game

Name	Type	Description
provider	string	Game provider name by GoldenGateX.
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.
gameId	string	id of the game provided by GoldenGateX.(optional) (This parameter only applies to specific slot games. If it is an empty string, it can be ignored.)
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.
gameName	string	Game name provided by GoldenGateX.
slug	string	Game slug provided by GoldenGateX.
thumbnail	string	Game thumbnail url.
updatedAt	string	Game updated datetime.
isNew	boolean	True - if the game new or false in other case
underMaintenance	boolean	False - if the game runs or false in other case

Example of HTTP request:

```
POST /api/v2/games/list HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "vendorCode": "casino-evolution",
        "language": "en"
}
```

Example of JSON response:

```
{
        "success": true,
        "message": [
                {
                        "provider": "Evolution",
                        "vendorCode": "casino-evolution",
                        "gameId": "",
                        "gameCode": "lobby",
                        "gameName": "Lobby",
                        "slug": "evolution-lobby",
                        "thumbnail":
"https://www.evolution.com/wp-content/uploads/2022/01/b2c_b2b_dual_play_pid_5.jpg",
                        "updatedAt": "2024-05-20T10:36:40.9800688Z",
                        "isNew": false,
                        "underMaintenance": false
                },
                ...
                {
                        "provider": "Evolution",
                        "vendorCode": "casino-evolution",
                        "gameId": "",
                        "gameCode": "LightningDice001",
                        "gameName": "Lightning Dice",
                        "slug": "evolution-lightningdice",
                        "thumbnail":
"https://www.evolution.com/wp-content/uploads/2022/01/lightning_dice_pid_10.jpg",
                        "updatedAt": "2024-01-10T01:18:18.043",
                        "isNew": false,
                        "underMaintenance": false
                }
        ],
        "errorCode": 0
}
```

2.4 Mini Game List

Using this method Casino Operator will get the list of mini games made by GoldenGateX available for the integration.

Request path: **GET** */games/mini/list*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	List of Game	The list of games.(see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: Game

Name	Type	Description
provider	string	Game provider name by GoldenGateX.
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.
gameId	string	id of the game provided by GoldenGateX.(optional) (This parameter only applies to specific slot games. If it is an empty string, it can be ignored.)
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.
gameName	string	Game name provided by GoldenGateX.
slug	string	Game slug provided by GoldenGateX.
thumbnail	string	Game thumbnail url.
isNew	boolean	True - if the game new or false in other case
underMaintenance	boolean	False - if the game runs or false in other case

Example of HTTP request:

```
GET /api/v2/games/mini/list HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
```

Example of JSON response:

```
{
        "message": [
                {
                        "provider": "GoldenGateX",
                        "vendorCode": "mini-crash",
                        "gameId": "",
                        "gameCode": "mini-crash",
                        "gameName": "Crash",
                        "slug": "crash",
                        "thumbnail": "https://fairvsgames.com/images/goldengatex-original/crash.png",
                        "isNew": false,
                        "underMaintenance": false
                },
                ...
                {
                        "provider": "GoldenGateX",
                        "vendorCode": "mini-keno",
                        "gameId": "",
                        "gameCode": "mini-keno",
                        "gameName": "Keno",
                        "slug": "keno",
                        "thumbnail": "https://fairvsgames.com/images/goldengatex-original/keno.png",
                        "isNew": false,
                        "underMaintenance": false
                }
        ],
        "errorCode": 0,
        "success": true
}
```

2.5 Game Detail

Using this method Casino Operator will get the detail of game available for the integration.

Request path: **POST** */game/detail*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.	Required
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	Game	The list of games.(see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: Game

Name	Type	Description
provider	string	Game provider name by GoldenGateX.
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.
gameId	string	id of the game provided by GoldenGateX.(optional) (This parameter only applies to specific slot games. If it is an empty string, it can be ignored.)
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.
gameName	string	Game name provided by GoldenGateX.
slug	string	Game slug provided by GoldenGateX.
thumbnail	string	Game thumbnail url.
updatedAt	string	Game updated datetime.
isNew	boolean	True - if the game new or false in other case
underMaintenance	boolean	False - if the game runs or false in other case

Example of HTTP request:

```
POST /api/v2/game/detail HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "vendorCode": "casino-evolution",
        "gameCode": "MonBigBaller0001"
}
```

Example of JSON response:

```
{
        "success": true,
        "message": {
                "provider": "Evolution",
                "vendorCode": "casino-evolution",
                "gameId": "",
                "gameCode": "MonBigBaller0001",
                "gameName": "MONOPOLY Big Baller",
                "slug": "evolution-monopolybigballer",
                "thumbnail":
"https://www.evolution.com/wp-content/uploads/2022/11/monopoly_big_baller_square_thumbn
ail_700x700_2022_07_01.jpg",
                "updatedAt": "2024-05-20T13:25:40.2246897Z",
                "isNew": false,
                "underMaintenance": false
        },
        "errorCode": 0
}
```

2.6 Get Launch URL

Using this method Casino Operator will get the launch url of the game.

Request path: **POST** */game/launch-url*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.	Required
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.	Required
userCode	string	Symbolic unique identifier of the user provided by Casino Operator.	Required
language	string	Language code. Ie. en, th,…	Required
lobbyUrl	string	Lobby url of the game website. Needed for redirection when some game providers close the game. Ex) Hacksaw slot game	Optional

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	string	This is the launch url of the game.
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/game/launch-url HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "vendorCode": "casino-evolution",
        "gameCode": "MonBigBaller0001",
        "userCode": "test",
        "language": "en",
        "lobbyUrl": "https://test.com",
}
```

Example of JSON response:

```
{
        "success": true,
        "message": "https://evolution.dcyqv8f2id.com/entry?jsessionid=72fa9603a8ef26381a112d86&lang=en&table=MonBigBaller0001",
        "errorCode": 0
}
```

~~2.7 Get Betting History (by date)~~ (deprecated)

Using this method Casino Operator will get the list of betting history by date(max **2000** records per page).

It is possible to get histories up to **7** days ago.

Currently the maximum time span from startDate to endDate is **8** hours.

Request path: **POST** */betting/history/by-date*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.	Optional
startDate	string	Start date time of the history. (UTC+0 time) Ex) ”yyyy-mm-dd” or ”yyyy-mm-ddThh:mm:ss”	Required
endDate	string	End date time of the history. (UTC+0 time) Ex) ”yyyy-mm-dd” or ”yyyy-mm-ddThh:mm:ss”	Required
page	int	Page number of the betting history	Required
perPage	int	History count per page (max 2000)	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	HistoryResponse	HistoryResponse (see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: HistoryResponse

Name	Type	Description
total	int	Total number of histories.
currentPage	int	The current page number.
perPage	int	The number of histories returned per page.
histories	List of History	The list of history.(see data type description below)

Type: History

Name	Type	Description
id	int	History Id by GoldenGateX.
userCode	string	Symbolic unique identifier of the user provided by Casino Operator.
roundId	string	Round Id of the game history.
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.
betAmount	decimal	Amount of the bet. Example: '1.00'.
winAmount	decimal	Amount of the win. Example: '1.00'.
beforeBalance	decimal	Balance before betting.
afterBalance	decimal	Balance after betting.
detail	string	Betting detail
status	int	Betting status. 0: Unfinished, 1: Finished, 2: Canceled
createdAt	timestamp	Betting created time.
updatedAt	timestamp	Betting updated time.

Example of HTTP request:

```
POST /api/v2/betting/history/by-date HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "vendorCode": "slot-pgsoft",
        "startDate": "2024-05-10T01:00:00",
        "endDate": "2024-05-10T09:00:00",
        "page": 1,
        "perPage": 2000
}
```

Example of JSON response:

```
{
    "success": true,
    "errorCode": 0,
    "message": {
        "total": 747,
        "currentPage": 1,
        "perPage": 2000,
        "startDate": "2024-04-22T05:28:05",
        "endDate": "2024-04-22T13:28:05",
        "histories": [
            {
                "id": 4,
                "userCode": "testuser",
                "roundId": "1713765639102",
                "gameCode": "vswaysmorient",
                "vendorCode": "mini-crash",
                "betAmount": 100.00,
                "winAmount": 76.00,
                "beforeBalance": 1000000.00,
                "afterBalance": 999976.00,
                "detail": "",
                "status": 1,
                "createdAt": 1713736839,
                "updatedAt": 1713736839
            },
            …
            {
                "id": 750,
                "userCode": "testuser",
                "roundId": "1713766466898",
                "gameCode": "vs10txbigbass",
                "vendorCode": "slot-pragmatic",
                "betAmount": 50.00,
                "winAmount": 10.00,
                "beforeBalance": 996287.00,
                "afterBalance": 996247.00,
                "detail": "",
                "status": 1,
                "createdAt": 1713737666,
                "updatedAt": 1713737667
            }
        ]
    }
}
```

2.8 Get Betting History(by Id)

Using this method Casino Operator will get the detail of betting history by id.

Request path: **POST** */betting/history/by-id*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
id	int	History id	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	History	The detail of history.(see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: History

Name	Type	Description
id	int	History Id by GoldenGateX.
userCode	string	Symbolic unique identifier of the user provided by Casino Operator.
roundId	string	Round Id of the game history.
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.
gameName	string	Game Name
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.
betAmount	decimal	Amount of the bet. Example: '1.00'.
winAmount	decimal	Amount of the win. Example: '1.00'.
beforeBalance	decimal	Balance before betting.
afterBalance	decimal	Balance after betting.
detail	string	Betting detail
status	int	Betting status. 0: Unfinished, 1: Finished, 2: Canceled
createdAt	timestamp	Betting created time.
updatedAt	timestamp	Betting updated time.

Example of HTTP request:

```
POST /api/v2/betting/history/by-id HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "id": 5
}
```

Example of JSON response:

```
{
        "success": true,
        "errorCode": 0,
        "message": {
                "id": 5,
                "userCode": "test",
                "roundId": "1713765639547",
                "gameCode": "vswaysmorient",
                "vendorCode": "slot-pragmatic",
                "betAmount": 100.00,
                "winAmount": 0.00,
                "beforeBalance": 999976.00,
                "afterBalance": 999876.00,
                "detail": "",
                "status": 1,
                "createdAt": 1713765639,
                "updatedAt": 1713765639
        }
}
```

2.9 Get Transaction History(by Id)

Using this method Casino Operator will get the detail transaction history by history id.

Request path: **POST** */transaction/history/by-id*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
id	int	History id	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	List of Transaction	The list of histories.(see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: Transaction

Name	Type	Description
id	int	Transaction id by GoldenGateX.
userCode	string	Symbolic unique identifier of the user provided by Casino Operator.
roundId	string	Round Id of the game history.
wagerId	int	History id.
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.
amount	decimal	Change balance. Example: '1.00'.
status	Int	0:success,1:fail
isFinished	boolean	This value indicates whether the round has ended. If you receive another round message after receiving this transaction, please reply with an error code ‘INVALID_TRANSACTION’.
isCanceled	boolean	This value indicates whether the bet has been canceled.
beforeBalance	decimal	Balance before transaction.
detail	string	Transaction detail
createdAt	timestamp	Betting created time.
updatedAt	timestamp	Betting updated time.

Example of HTTP request:

```
POST /api/v2/transaction/history/by-id HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "id": "99"
}
```

Example of JSON response:

```
{
        "success": true,
        "errorCode": 0,
        "message": [
        {
                "id": 5,
                "userCode": "test",
                "roundId": "1713765639547",
                "wagerId": "17",
                "gameCode": "vswaysmorient",
                "vendorCode": "slot-pragmatic",
                "amount": -100.00,
                “status”: 0,
                “isFinished”: false,
                “isCanceled”: false,
                "beforeBalance": 999976.00,
                "detail": "",
                "createdAt": 1713765639,
                "updatedAt": 1713765639
        }
]
}
```

2.10 Get Agent Balance

Using this method Casino Operator will get agent balance.

Request path: **GET** */agent/balance*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	decimal	User balance
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
GET /api/v2/agent/balance HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU…..yC9dBS_kwww
```

Example of JSON response:

```
{
        "message": 1000,
        "errorCode": 0,
        "success": true
}
```

2.11 Create User (Transfer API)

Using this method Casino Operator will create user for transfer method.

Request path: **POST** */user/create*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
userCode	string	userCode from Casino operator	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	string	Description of the response
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/user/create HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU…..yC9dBS_kwww
Request Body
{
        "userCode": "testuser1"
}
```

Example of JSON response:

```
{
        "success": true,
        "errorCode": 0
}
```

2.12 Balance (Transfer API)

Using this method Casino Operator will get player balance for transfer method.

Request path: **POST** */user/balance*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
userCode	string	userCode from Casino operator	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	decimal	Balance of the player
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/user/balance HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "userCode": "testuser1"
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 1000.00,
        "errorCode": 0
}
```

2.13 Deposit(Balance Transfer API)

Using this method Casino Operator will deposit player balance.

Request path: **POST** */user/deposit*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
userCode	string	userCode from Casino operator	Required
balance	decimal	Amount to deposit.	Required
orderNo	string	This is identifier for deposit.	Optional
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX. (For the separated wallet this field should be required.)	Optional

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	decimal	User balance
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/user/deposit HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
Request Body
{
        "userCode": "testuser1",
        "balance" : 1000.00,
        "orderNo" : "EF232FAD43"
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 1000,
        "errorCode": 0
}
```

2.14 Withdraw(Balance Transfer API)

Using this method Casino Operator will withdraw user balance.

Request path: **POST** */user/withdraw*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
userCode	string	userCode from Casino operator	Required
balance	decimal	Amount to deposit.	Required
orderNo	string	This is identifier for withdraw.	Optional
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX. (For the separated wallet this field should be required.)	Optional

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	decimal	User balance
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/user/withdraw HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
{
        "userCode": "testuser",
        "balance" : 1000.00,
        "orderNo" : "EF232FAD42"
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 1000,
        "errorCode": 0
}
```

2.15 Withdraw All(Balance Transfer API)

Using this method Casino Operator will withdraw all player balance.

Request path: **POST** */user/withdraw-all*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
userCode	string	userCode from Casino operator	Required
vendorCode	string	Symblic unique identifier of the vendor provided by GoldenGateX. (For the separated walltet this field should be required.)	Optional

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	decimal	Withdraw balance
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/user/withdraw-all HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
{
        "userCode": "testuser",
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 1000,
        "errorCode": 0
}
```

2.16 Get User Balance Log(Balance Transfer API)

Using this method Casino Operator will withdraw all player balance.

Request path: **POST** */user/balance-history*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
orderNo	string	orderNo from Casino operator	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	UserBalanceHistory	The detail of history.(see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: UserBalanceHistory

Name	Type	Description
id	int	Transaction id by GoldenGateX.
userCode	string	userCode from Casino operator.
amount	decimal	Request balance. Example: '1.00'.
type	int	1:deposit,2:withdraw
agentBeforeBalance	decimal	Agent balance before transaction
userBeforeBalance	decimal	User balance before transaction

Example of HTTP request:

```
POST /api/v2/user/balance-history HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
{
        "orderNo": "EF232FAD42",
}
```

Example of JSON response:

```
{
        "success": true,
        "message": {
                "userCode": "testuser1",
                "amount": 1000.00,
                "agentBeforeBalance": 10197667478346.52,
                "userBeforeBalance": 3000.00,
                "type": 1,
                "createdAt": 1724755309
        },
        "errorCode": 0
}
```

2.17 Set User RTP

Using this method Casino Operator will set user's RTP.

Request path: **POST** */game/user/set-rtp*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symblic unique identifier of the vendor provided by GoldenGateX.	Required
userCode	string	userCode from Casino operator.	Required
rtp	int	Number between 30 and 99	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/user/set-rtp HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
{
	"vendorCode": "slot-pragmatic",
	"userCode": " testuser1",
	"rtp": 90
}
```

Example of JSON response:

```
{
	"success": true,
	"errorCode": 0
}
```

2.18 Get User RTP

Using this method Casino Operator will get user’s RTP.
Request path: **POST** */game/user/get-rtp*
Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symblic unique identifier of the vendor provided by GoldenGateX.	Required
userCode	string	userCode from Casino operator.	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	int	User’s RTP
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/user/get-rtp HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU…..yC9dBS_kwww
{
        "vendorCode": "slot-pragmatic",
        "userCode": " testuser1"
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 85
        "errorCode": 0
}
```

2.19 Reset Users RTP

Using this method Casino Operator will set all user’s RTP.
Request path: **POST** */game/users/reset-rtp*
Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symblic unique identifier of the vendor provided by GoldenGateX.	Required
rtp	int	RTP for all users you are trying to set	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	int	All user’s RTP
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/users/reset-rtp HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU.....yC9dBS_kwww
{
        "vendorCode": "slot-pragmatic",
        "rtp": 85
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 85
        "errorCode": 0
}
```

2.20 Get Betting History V2 (by date)

Using this method Casino Operator will get the list of betting history by date(max **5000** records).
It is possible to get histories up to **7** days ago.
This API allows a maximum of **one request per second**. Exceeding this limit may result in temporary throttling or request rejection.
Request path: **POST** */betting/history/by-date-v2*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.	Optional
startDate	string	Start date time of the history. (UTC+0 time) Ex) ”yyyy-mm-dd” or ”yyyy-mm-ddThh:mm:ss”	Required
limit	int	History count (max 5000)	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	HistoryResponseV2	HistoryResponseV2 (see data type description below)
errorCode	int	0 - if the request was successfully processed or error code in other case.

Type: HistoryResponseV2

Name	Type	Description
nextStartDate	string	Start date time of the history for next request. (UTC+0 time) Ex) ”yyyy-mm-dd” or ”yyyy-mm-ddThh:mm:ss”
limit	int	Requested History count.
histories	List of History	The list of history.(see data type description below)

Type: History

Name	Type	Description
id	int	History Id by GoldenGateX.
userCode	string	Symbolic unique identifier of the user provided by Casino Operator.
roundId	string	Round Id of the game history.
gameCode	string	Symbolic unique identifier of the game provided by GoldenGateX.
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX.
betAmount	decimal	Amount of the bet. Example: '1.00'.
winAmount	decimal	Amount of the win. Example: '1.00'.
beforeBalance	decimal	Balance before betting.
afterBalance	decimal	Balance after betting.
detail	string	Betting detail
status	int	Betting status. 0: Unfinished, 1: Finished, 2: Canceled
createdAt	timestamp	Betting created time.
updatedAt	timestamp	Betting updated time.

Example of HTTP request:

```
POST /api/v2/betting/history/by-date HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU…..yC9dBS_kwww
Request Body
{
        "vendorCode": "slot-pgsoft",
        "startDate": "2024-04-21T01:00:00",
        "limit": 5000
}
```

Example of JSON response:

```
{
        "success": true,
        "errorCode": 0,
        "message": {
                "limit": 5000,
                "nextStartDate": "2024-4-21T10:14:27",
                "histories": [
                        {
                                "id": 4,
                                "userCode": "testuser",
                                "roundId": "1713765639102",
                                "gameCode": "vswaysmorient",
                                "vendorCode": "mini-crash",
                                "betAmount": 100.00,
                                "winAmount": 76.00,
                                "beforeBalance": 1000000.00,
                                "afterBalance": 999976.00,
                                "detail": "",
                                "status": 1,
                                "createdAt": 1713736839,
                                "updatedAt": 1713736839
                        },
                        …
                        {
                                "id": 750,
                                "userCode": "testuser",
                                "roundId": "1713766466898",
                                "gameCode": "vs10txbigbass",
                                "vendorCode": "slot-pragmatic",
                                "betAmount": 50.00,
                                "winAmount": 10.00,
                                "beforeBalance": 996287.00,
                                "afterBalance": 996247.00,
                                "detail": "",
                                "status": 1,
                                "createdAt": 1713737666,
                                "updatedAt": 1713737667
                        }
                ]
        }
}
```

2.21 Get Betting Detail Page URL

Using this method Casino Operator will get the detail page url of the betting.

Request path: **POST** */betting/history/detail*

Request Headers

Name	Type	Description	Status
Authorization	string	Bearer {token} Here operator can use token generated from createToken request.	Required

Request parameters

Name	Type	Description	Status
Id	Int	History Id by GoldenGateX.	Required
language	string	Language code. Ie. en, zh,…	Optional

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	string	This is the detail page url of the betting.
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/v2/betting/history/detail HTTP/2
Host: xxx.com
content-type: application/json
authorization: Bearer eyJhbGciOiJIUzU…..yC9dBS_kwww
Request Body
{
        "id": 20000,
        "language": "en"
}
```

Example of JSON response:

```
{
        "success": true,
        "message": " https://m32unew6.com/gs2c/gameDetail?
token=06BBAC92439C373931352CD7273D6C6F979B674171FB18777E73AD2572D01540247CF163
&playSessionId=44388079111140&lang=en ",
        "errorCode": 0
}
```

III Seamless Wallet API (Operator API)

Operator should provide the Seamless Wallet Integration API on their side. GoldenGateX will call the methods when players make a bet or get a win and their balance should be updated.

Method/Section	Description	Status
Balance	Returns the balance of a player.	Required
Transaction	Returns the balance of a player.	Required

3.1 Balance

Using this method Casino Operator will return user balance.

* To generate an authentication header, please check this link https://www.debugbear.com/basic-auth-header-generator

Request path: **POST** */api/balance*

Request Headers

Name	Type	Description	Status
Authorization	string	Basic {base 64 encoding of a clientId and clientSecret separated by a colon}	Required

Request parameters

Name	Type	Description	Status
userCode	string	userCode from Casino operator	Required

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	decimal	User balance
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/balance HTTP/2
Host: xxx.com
authorization: Basic dGVzdE.....WNyZXQ=
content-type: application/json
{
        "userCode": "testuser",
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 1000,
        "errorCode": 0
}
```

3.2 Transaction

Using this method a GoldenGateX system will notify Casino Operator about game transaction.
* To generate an authentication header, please check this link
https://www.debugbear.com/basic-auth-header-generator

Request path: **POST** */api/transaction*
Request Headers

Name	Type	Description	Status
Authorization	string	Basic {base 64 encoding of a clientId and clientSecret separated by a colon}	Required

Request parameters

Name	Type	Description
userCode	string	userCode from Casino operator
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX
gameCode	string	Symbolic unique identifier of the board game provided by GoldenGateX.
historyId	int	Game History Id. This is unique for game History. This is used to check game and transaction history.
roundId	string	Game round identity. This is unique for game round.
gameType	int	Game type 1: Casino, 2: Slot, 3:Other(Original games and Board Games), 4:Fishing
transactionCode	string	This is a identity for transaction. If these values are duplicates, respond with an error code 'DUPLICATE_TRANSACTION'.
isFinished	boolean	This value indicates whether the round has ended. If you receive another round message after receiving this transaction, please reply with an error code 'INVALID_TRANSACTION'.
isCanceled	boolean	This value indicates whether the bet has been canceled.
amount	decimal	transaction amount if this is a betting transaction(*old debit), this becomes negative. And if this is win transaction(*old credit), this becomes positive.
detail	string	Detail of transaction description
createdAt	Datetime(utc)	This is the date the transaction was created.

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case
message	decimal	User balance after transaction
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/transaction HTTP/2
Host: xxx.com
authorization: Basic dGVzdE.....WNyZXQ=
content-type: application/json
{
        "userCode": "testuser",
        "vendorCode": "slot-pragmatic",
        "gameCode": "vs20doghouse",
        “historyId”: ”2228221”,
        “roundId”: ”178482632282”,
        “gameType”: 2,
        “transactionCode”:”a2c333bb67”,
        “isFinished”: false,
        “isCanceled”: false,
        "amount": -1000,
        "detail":"{}",
        "createdAt":"2024-06-15 12:0:0",
}
```

Example of JSON response:

```
{
        "success": true,
        "message": 1000,
        "errorCode": 0
}
```

3.3 Batch Transactions

Using this method a GoldenGateX system will notify Casino Operator about game transaction.
* To generate an authentication header, please check this link
https://www.debugbear.com/basic-auth-header-generator

Request path: **POST** */api/batch-transactions*

Request Headers

Name	Type	Description	Status
Authorization	string	Basic {base 64 encoding of a clientId and clientSecret separated by a colon}	Required

Request parameters

Name	Type	Description
userCode	string	userCode from Casino operator
transactions	List of TransactionRequest	The list of TransactionRequest. (Please confirm data type of TransactionRequest.)

Type: TransactionRequest

Name	Type	Description
userCode	string	userCode from Casino operator
vendorCode	string	Symbolic unique identifier of the vendor provided by GoldenGateX
gameCode	string	Symbolic unique identifier of the board game provided by GoldenGateX.
historyId	int	Game History Id. This is unique for game History. This is used to check game and transaction history.
roundId	string	Game round identity. This is unique for game round.
gameType	int	Game type 1: Casino, 2: Slot, 3:Other(Original games and Board Games), 4:Fishing
transactionCode	string	This is an identity for a transaction. If these values are duplicates, respond with an error code 'DUPLICATE_TRANSACTION'.
isFinished	boolean	This value indicates whether the round has ended. If you receive another round message after receiving this transaction, please reply with an error code 'INVALID_TRANSACTION'.
isCanceled	boolean	This value indicates whether the bet has been cancelled.
amount	decimal	Transaction amount If this is a betting transaction(*old debit), this becomes negative. And if this is a win transaction(*old credit), this becomes positive.
detail	string	Detail of transaction description
createdAt	Datetime(UTC)	This is the date the transaction was created.

Response parameters

Name	Type	Description
success	boolean	True - if the request was successfully processed or false in other case

message	decimal	User balance after transaction
errorCode	int	0 - if the request was successfully processed or error code in other case.

Example of HTTP request:

```
POST /api/batch-transactions HTTP/2
Host: xxx.com
authorization: Basic dGVzdE.....WNyZXQ=
content-type: application/json
{
	"userCode": "testuser",
	"transactions": [
		{
			"userCode": "testuser",
			"vendorCode": "slot-pragmatic",
			"gameCode": "vs20doghouse",
			"historyId": "2228221",
			"roundId": "178482632282",
			"gameType": 2,
			"transactionCode": "a2c333bb67",
			"isFinished": false,
			"isCanceled”: false,
			"amount": -1000,
			"detail":"{}",
			"createdAt": "2024-06-15 12:0:0",
		},
		…
		{
			"userCode": "testuser",
			"vendorCode": "slot-pragmatic",
			"gameCode": "vs20doghouse",
			"historyId": "2228221",
			"roundId": "178482632282",
			"gameType": 2,
			"transactionCode": "a2c333bb67",
			"isFinished": false,
			"isCanceled”: false,
			"amount": -1000,
			"detail":"{}",
			"createdAt": "2024-06-15 12:0:0"
		}
	]
}
```

Example of JSON response:

```
{
	"success": true,
	"message": 1000,
	"errorCode": 0
}
```

Error Codes

Name	Value	Description
NO_ERROR	0	No errors
USER_ALREADY_EXISTS	1	User account you are trying to create already exists in the system.
USER_DOES_NOT_EXIST	2	Unable to find a user account with the provided credentials or identifier.
INSUFFICIENT_AGENT_BALANCE	3	Agent does not have enough balance in the account to complete the requested action or transaction.
INSUFFICIENT_USER_BALANCE	4	User does not have enough balance in the account to complete the requested action or transaction.
NO_BETTING_LOG_EXIST	5	Unable to find any betting or transaction logs for the request.
DUPLICATE_TRANSACTION	6	Transaction Code is duplicated.
INVALID_TRANSACTION	7	This wager is already finished.
BALANCE_LOG_DOES_NOT_EXIST	8	User balance log does not exist.
VENDOR_IS_UNDER_MAINTENANCE	9	The vendor is under maintenance.
GAME_IS_UNDER_MAINTENANCE	10	The game is under maintenance.
DEPRECATED_ENDPOINT	20	Endpoint is deprecated.
BAD_REQUEST	400	The request sent to the system is invalid or incomplete, and cannot be processed as expected.
UNAUTHORIZED	401	User or agent does not have the necessary credentials to perform the requested action or access the requested resource.
UNKNOWN_SERVER_ERROR	500	Unexpected or unidentified error has occurred on the server, preventing the system from processing the request as expected.

Language Codes

Name	Description
ar	Arabic
bg	Bulgarian
ca	Catalan, Valencian
cs	Czech
da	Danish
de	German
el	Greek
en	English
es	Spanish, Castilian
et	Estonian
fi	Finnish
fr	French
he	Hebrew
hi	Hindi
hr	Croatian
hu	Hungarian
hy	Armenian
id	Indonesian
it	Italian
ja	Japanese
ka	Georgian
ko	Korean
lt	Lithuanian
lv	Latvian
mn	Mongolian
ms	Malay
nl	Dutch
pl	Polish
pt	Portuguese
ro	Romanian
ru	Russian
sk	Slovak
sl	Slovene
sq	Albanian
sv	Swedish
th	Thai
tr	Turkish
uk	Ukrainian
vi	Vietnamese
ar	Arabic
bg	Bulgarian
ca	Catalan, Valencian
cs	Czech
da	Danish
de	German
el	Greek
en	English
es	Spanish, Castilian
et	Estonian
fr	French
he	Hebrew
hi	Hindi

hr	Croatian
hu	Hungarian
hy	Armenian
id	Indonesian
it	Italian
ja	Japanese
ka	Georgian
ko	Korean
lt	Lithuanian
lv	Latvian
mn	Mongolian
ms	Malay
nl	Dutch
pl	Polish
pt	Portuguese
ro	Romanian
ru	Russian
sk	Slovak
sl	Slovene
sq	Albanian
sv	Swedish
th	Thai
tr	Turkish
uk	Ukrainian
vi	Vietnamese
zh	Chinese

Currency Codes

Currency Symbol	Currency Code	Description
$	USD	US Dollar
฿	THB	Thai Baht
₹	INR	Indian Rupee
₩	KRW	South Korean Won
JP¥	JPY	Japanese Yen
₮	MNT	Mongolian Tugrik
R$	BRL	Brazilian Real
CN¥	CNY	Chinese Yuan
đ	VND	Vietnamese Dong
₺	TRY	Turkish Lira
Rp	IDR	Indonesian Rupiah
₱	PHP	Philippine Peso
€	EUR	Euro
$AR	ARS	Argentine Peso
R	ZAR	South African Rand
DT	TND	Tunisia Dinar
₦	NGN	Nigerian naira
S/	PEN	Peruvian sol
$U	UYU	Uruguayan Peso
Col$	COP	Colombian peso
Bs	BOB	Bolivian boliviano
CLP	CLP	Chilean peso
Mex$	MXN	Mexican peso
RM	MYR	Malaysian ringgit
৳	BDT	Bangladeshi Takas
AU$	AUD	Australian dollar
K	MMK	Myanmar kyat
₭	LAK	Lao kip
₲	PYG	Paraguayan Guarani
NT$	TWD	New Taiwan dollar
₽	RUB	Russian Ruble
Rs	PKR	Pakistani Rupee
£	EGP	Egyptian pound
Лв	UZS	Uzbekistani Som