

# GamMatrix Server API

Repository changes

Author	Date	Comments
@ Anca Neagu	22.07.2016	<p>Added new ServerAPI methods:</p> <p>ActivatePlayer, ProceedWithdraw, RollbackWithdraw, CheckDepositTrans,</p> <p>SetUserRgLossLimit, SetUserWageringLimit, SetUserSessionLimit,</p> <p>GetUserRgLossLimit, GetUserRgWageringLimit, GetUserRgSessionLimit,</p> <p>RemoveUserRgWageringLimit, RemoveUserRgSessionLimit, RemoveUserRgDepositLimit, RemoveUserRgLossLimit.</p> <p>Updated ServerAPI methods: Deposit, Withdraw.</p>
@ Anca Neagu	23.08.2016	<p>Updated Transfer method with new parameter.</p> <p>Updated description on "ExpiryDate" variable for SetUserRgSelfExclusion and SetUserRgCoolOff methods.</p>
@ Anca Neagu	18.10.2016	Added new ServerAPI method: ApplicableBonusList
@ Anca Neagu	02.11.2016	Corrected method ApplicableBonusList with the supported bonus types.
@ Anca Neagu	10.02.2017	<p>Updated lovation requests; if lovation is enabled, the parameter will not become mandatory (will still be optional).</p> <p>Added new ServerAPI method: GetAccountBalanceForMultipleUsers <i>VendorID+Wallet"</i></p>
@ Anca Neagu	15.02.2017	Added clarifications for method UpdateUserDetails, birthDate parameter
@ Anca Neagu	31.03.2017	Added change for <i>LoginUser</i> method which now supports login by email as well.
@ Anca Neagu	15.06.2017	Corrected Transfer method. Updated error codes with more values.
@ Anca Neagu	11.07.2017	Added new ServerAPI method: <i>AgentResetPlayerPassword</i>
@ Anca Neagu	22.11.2017	Added minor details on <i>Transfer</i> method.
@ Daniel Danaila	10.08.2018	<p><b>Updated examples for methods:</b> <i>AgentImpersonateUser, AgentTransfer, ApplyAffiliateChipBonus, ApplyUserBonus, ApplyUserExternalBonus, ApproveDepositLimit, AssignUserRole, AssignUserRoleWithExpiration, CancelDepositLimit, ChangeUserPassword, ForfeitUserCasinoBonus, GenerateUserHash, GetAvailableRealityCheckValues, GetAvailableBonusList, GetCoolOffSettings, GetSelfExclusionSettings, GetUserAccounts, GetUserBonusDetails, GetUserDetails, GetUserPaycards, GetUserMetadata, GetUserPendingWithdrawals, IsActiveUserSession, IsAliasAvailable, IsEmailAvailable, IsUserHashValid, IsUserNameAvailable, IsValidCredentials, RegisterPayCard, LogoutUser, ManualDeposit, ManualWithdraw, RegisterPayCard, RegisterUser, RemoveuserRole, SetUserForgotPassInfo, SetUserMetadata, SetUserRealityCheck, SetUserRgCoolOff, SetUserRgSelfExclusion, Transfer, UpdatePayCardStatus, UpdateUserDetails</i> and <i>LoginUser</i>.</p> <p><b>Added parameters and updated examples:</b> <i>errorData, errorCode, errorDetails, errorMessage, logId, success, timestamp</i> and <i>version</i> for method <i>ApplicableBonusList</i>.</p> <p><b>Added parameters and updated examples:</b> <i>debitPayCardVendorId</i> and <i>applyBonusVendorID</i> for method <i>CheckDepositTrans</i>. <i>Sessionid</i> and <i>creditAccountId</i> parameters are no longer mandatory.</p> <p><b>Added parameters and updated examples:</b> <i>firstName, lastName, phone</i> and <i>zip</i> to method <i>CheckUserExists</i>. <i>Birthdate</i> and <i>email</i> parameters are no longer mandatory.</p> <p><b>Added parameter and updated examples:</b> <i>paymentMethod</i> to <i>Deposit</i> method. <i>SessionId</i> no longer mandatory for the request.</p> <p><b>Added new methods:</b> <i>DepositAndPlay, LoginAndPlay, DomainHealthCheck</i> (in Monitoring), <i>GetBalance</i>.</p>

@ Daniel Danaila	20.08.2018	<p><b>Updated request parameters:</b> sessionId for method <i>GetBalance</i>, sessionId, period, amount and currency for method <i>SetDepositLimit</i>.</p> <p><b>Updated method information:</b> password parameter for <i>LoginUser</i> method.</p> <p><b>Added new methods:</b> <i>GetDepositMaxAmount</i>, <i>GetUserAssignedRoles</i>, <i>SearchUsers</i>, <i>GetUserPromotions</i>, <i>SetUserPromotion</i>.</p> <p><b>Added parameters and updated examples:</b> paymentMethod parameter for <i>Withdraw</i> method.</p>
@ Daniel Danaila	21.08.2018	<p><b>Updated examples:</b> <i>GetUsersRoles</i>.</p> <p><b>Added new method:</b> <i>GlobalHealthCheck</i> to Monitoring section.</p>
@ Daniel Danaila	12.09.2018	<b>Added parameters and updated examples:</b> language and isMobile optional parameters added to methods <i>DepositAndPlay</i> and <i>LoginAndPlay</i> .
@ Daniel Danaila	05.10.2018	<b>Added parameters and updated example:</b> userId for method <i>SetUserRgSelfExclusion</i>
@ Daniel Danaila	23.10.2018	<b>Added parameters and updated examples:</b> sessionId for methods <i>LoginAndPlay</i> and <i>DepositAndPlay</i> .
@ Daniel Danaila	25.10.2018	<p><b>Added new section:</b> Deprecated Methods</p> <p>Moved <b>Bonus</b> section to Deprecated Methods</p>
@ Daniel Danaila	03.12.2018	<p><b>Corrected:</b> <i>Deposit</i>, <i>CheckDepositTrans</i> and <i>Withdraw</i> methods missing userId parameter. Updated the description for sessionId and examples.</p> <p><b>Updated Examples:</b> <i>CheckUserExists</i> method</p>
@ Daniel Danaila	17/12/2018	<b>Update method and example:</b> preTransId parameter added to <i>RollbackWithdraw</i> method
@ Daniel Danaila	19/12/2018	<b>Updated examples:</b> <i>GetUserRgDepositLimit</i> , <i>GetUserRgSelfExclusion</i>
@ Daniel Danaila	28/01/2019	<b>Updated methods and examples:</b> personalId response parameter (string type, optional) added to <i>RegisterUser</i> , <i>UpdateUserDetails</i> , <i>CheckUserExists</i> , <i>GetUserDetails</i>
@ Daniel Danaila	15/02/2019	<p><b>Added parameters and updated examples:</b> registrationDate response parameter for <i>Login</i> method, userId request parameter for <i>UpdateUserDetails</i> and <i>SetUserRgCoolOff</i> methods.</p> <p><b>Added new section and method:</b> <i>GetUserLinkedPlayers</i> in <b>Linked Players Methods</b></p>
@ Daniel Danaila	20/02/2019	Updated paymentMethod parameter description for <i>Withdraw</i> and <i>Deposit</i> methods.
@ Daniel Danaila	27/03/2019	<b>Added new parameter:</b> transactionReference to <i>GetUserPendingWithdrawals</i> method
@ Daniel Danaila	11/04/2019	<b>Added new Responsible Gambling methods:</b> <i>SetRgCumulativeSessionLimit</i> , <i>GetRgCumulativeSessionLimits</i> , <i>RemoveRgCumulativeSessionLimit</i>
@ Daniel Danaila	03/05/2019	<b>Added new method to Payment section:</b> <i>GetUserTransactionHistory_RecentGaming</i>
@ Daniel Danaila	06/05/2019	<b>Added new methods to Payment section:</b> <i>GetUserTransactionHistory_Gaming</i> , <i>GetUserTransactionHistory_Payment</i> and <i>GetUserTransactionHistory_Vendor</i>
@ Daniel Danaila	17/05/2019	<b>Added new parameter and updated example:</b> approvalStatus for <i>GetUserPendingWithdrawals</i> method.
@ Daniel Danaila	20/06/2019	<b>Removed parameters:</b> IsFirstDepositBonusAvailableAndRequireTC parameter from <i>Deposit</i> method and isFirstDepositBonusTCAccepted parameter from <i>ProcessAsyncTrans</i> as they're no longer in use.
@ Daniel Danaila	11 Jul 2019	<b>Added new parameters and updated examples:</b> nationality, birthplace and birthCountryCode to methods <i>RegisterUser</i> , <i>UpdateUserDetails</i> and <i>GetUserDetails</i> .
@ Daniel Danaila	22/07/2019	<b>Added new request parameter and updated URL:</b> productCategory for <i>GetBalance</i> method..
@ Daniel Danaila	23/08/2019	<b>Moved ManualDeposit and ManualWithdraw methods to OBSOLETE section.</b>
@ Daniel Danaila	18/10/2019	<b>Added new method to User Account section:</b> <i>SetUserStatus</i> for blocking or unblocking a player's account.

@ Daniel Danaila	31/10/2019	<b>Added affiliateMarker parameter</b> to <code>UpdateUserDetails</code> method.
@ Daniel Danaila	28/11/2019	<b>Updated RegisterUser parameters:</b> phone and mobile minimum number of digits <b>lowered to 5 from 7</b> .
@ Daniel Danaila	12/02/2020	<b>Updated method: LoginUser - Use token instead of password</b>
@ Daniel Danaila	17/02/2020	<b>Added expiryDate parameter to method <code>GetUserRgSelfExclusion</code></b>

- Introduction
- 1) Pre-Registration
  - a) IsUserNameAvailable
    - Response variables
    - Response example
  - b) IsEmailAvailable
    - Response variables
    - Response example
  - c) IsAliasAvailable
    - Response variables
    - Response example
  - d) CheckUserExists
    - Request variables
    - Response variables
    - Request example
    - Response example (success)
    - Response example (user does not exist)
    - Response example (empty request)
- 2. Registration
  - a) RegisterUser
    - Request variables
    - Response variables
    - Request example
    - Response example
  - b) GetUserDetails
    - Response variables
    - Response example
  - c) UpdateUserDetails
    - Request variables
    - Response variables
    - Request example
    - Response example
  - d) ChangeUserPassword
    - Request variables
    - Response variables
    - Request example
    - Response example
  - e) SetUserForgotPassInfo
    - Request variables
    - Response variables
    - Request example
    - Response example
  - f) GenerateUserHash
    - Request variables
    - Response variables
    - Request example
    - Response example
  - g) IsUserHashValid
    - Response variables
    - Response example
  - h) ActivatePlayer

- Request variables
  - Response variables
  - Request example
  - Response example Successful response
  - Response example Failed response (invalid or expired hash)
- 3. Login
  - a) LoginUser
    - Request variables
    - Response variables
    - Request examples
    - Response example
  - b) LoginUser (with Iovation enabled)
    - Request variables
    - Response variables
    - Request example (only for POST methods)
    - Response example
    - Response variables
    - Response example
  - d) IsActiveUserSession
    - Response variables
    - Response example
  - e) IsValidCredentials
    - Response variables
    - Response example
- 4. User account
  - a) GetUserAccounts
    - Response variables
    - Response example
  - b) GetUsersRoles
    - Response variables
    - Response example
  - c) AssignUserRole
    - Request variables
    - Response variables
    - Request example
    - Response example
  - d) AssignUserRoleWithExpiration
    - Request variables
    - Response variables
    - Request example
    - Response example
  - e) RemoveUserRole
    - Request variables
    - Response variables
    - Request example
  - f) SetUserMetadata
    - Request variables
    - Response variables
    - Request example
    - Response example
  - g) GetUserMetadata
    - Response variables
    - Response example
  - h) GetUserAssignedRoles
    - Request variables
    - Response variables
    - Response example
  - i) SearchUsers
    - Request variables
    - Response variables
    - Request Example

- By personal id
  - By username
  - Response Example
- j) SetUserStatus
  - Request Variables
- Request Example
- Response Variables
- Response Example
- 5. Payment
  - a) RegisterPayCard
    - Request variables
    - Response variables
    - Request example
    - Response example
  - b) GetUserPayCards
    - Response variables
    - Response example
  - c) Deposit
    - Request variables
    - Response variables
    - Response example
  - d) Withdraw
    - Request variables
    - Response variables
    - Request example
    - Response example
  - e) ProceedWithdraw
    - Request variables
    - Response variables
    - Request example
    - Response example
  - f) RollbackWithdraw
    - Request variables
    - Response variables
    - Request example
    - Response example
    - Response example (preTransId and transactionReference both present)
  - g) ProcessAsyncTrans
    - Request variables
    - Response variables
    - Request example
    - Response example
  - h) GetUserTransactionHistory
    - Request variables
    - Response variables
    - Request example
    - Response example (No transactions)
    - Response example
  - i) GetUserPendingWithdrawals
    - Response variables
    - Response example
  - j) UpdatePayCardStatus
    - Request variables
    - Response variables
    - Request example
    - Response example
  - m) Transfer
    - Request variables
    - Request variables body
    - Response variables
    - Request example

- Response example (successful response)
- Response example (failed response - session not found)
- Response example (failed response - bonus code not authorized)
- Response example (failed response - bonus vendor not valid)
- Response example (failed response - currency rate is missing)
- n) GetUserTransactionList
  - Request variables
  - Response variables
  - Request example (only for POST methods)
  - Response example
- o) DepositAndPlay
  - Request variables
  - Response variables
  - Request example
  - Response example
- p) LoginAndPlay
  - Request variables
  - Response variables
  - Request example
  - Response example
- q) GetBalance
  - Request variables
  - Response variables
  - Response example
- r) GetDepositMaxAmount
  - Request variables
  - Response variables
  - Request example
  - Response example
  - Response if user is blocked:
  - Response if limit is reached:
- s) GetUserTransactionHistory\_RecentGaming
- t) GetUserTransactionHistory\_Gaming
- u) GetUserTransactionHistory\_Payment
- v) GetUserTransactionHistory\_Vendor
- 6. Responsible gambling
  - a) SetDepositLimit
    - Request variables
    - Response variables
    - Request example
    - Response example
  - b) GetUserRgDepositLimit
    - Response variables
    - Response example (no limits)
    - Response example
  - c) ApproveDepositLimit
    - Response variables
    - Response example
  - d) CancelDepositLimit
    - Response variables
    - Response example
  - e) SetUserRgSelfExclusion
    - Request variables
    - Response variables
    - Request example
    - Response example
  - f) GetUserRgSelfExclusion
    - Response variables
    - Response example (no self exclusion)
  - g) GetCoolOffSettings
    - Response variables

- Response example
- h) GetSelfExclusionSettings
  - Response variables
  - Response example
- i) SetUserRgCoolOff
  - Request variables
  - Response variables
  - Request example
  - Response example
- j) GetUserRgDepositLimitList
  - Response variables
  - Response example
- k) GetAvailableRealityCheckValues
  - Response variables
  - Request variables from URL
  - Response example
- l) GetUserRealityCheck
  - Request variables from the URL
  - Response variables
  - Response example
- m) SetUserRealityCheck
  - Request variables from the URL
  - Variables from the method (from the body)
  - Response variables
  - Request example (only for POST methods)
  - Response example
- n) CheckDepositTrans
  - Request variables
  - Response variables
  - Request example
  - Response example
- o) SetUserRgLossLimit
  - Response variables:
  - Response example
- p) SetUserRgWageringLimit
  - Response variables
  - Response example
- q) SetUserRgSessionLimit
  - Response variables
  - Response example
- r) GetUserRgLossLimit
  - Response variables
  - Response example
  - Response example (with scheduled update)
- s) GetUserRgWageringLimit
  - Response variables
  - Response example
  - Response example (with scheduled update)
- t) GetUserRgSessionLimit
  - Response variables
  - Response example
  - Response example (with scheduled update)
- u) RemoveUserRgWageringLimit
  - Request variables
  - Response variables
  - Request example
  - Response example
- v) RemoveUserRgSessionLimit
  - Request variables
  - Response variables
  - Request example

- Response example
- w) RemoveUserRgDepositLimit
  - Request variables
  - Response variables
  - Request example
  - Response example
- x) RemoveUserRgLossLimit
  - Request variables
  - Response variables
  - Request example
  - Response example
- y) SetRgCumulativeSessionLimit
  - Request variables
  - Request example:
  - Response variables:
  - Response example:
- z) GetRgCumulativeSessionLimits
  - Request variables:
  - Request example:
  - Response parameters:
  - Response example:
- aa) RemoveRgCumulativeSessionLimit
  - Request variables
  - Request example:
  - Response variables
  - Response example:
- 7. Partner Matrix 2
  - a) AgentTransfer
    - Request variables
    - Response variables
    - Request example
    - Response example
  - b) AgentImpersonateUser
    - Request variables
    - Response variables
    - Request example
    - Response example
  - c) GetAccountBalanceForMultipleUsers
    - Request variables
    - PayLoad
    - Response variables
    - Request example
    - Response example
  - d) AgentResetPlayerPassword
    - Request variables
    - Request example
    - Response example
- 8. Monitoring
  - a) DomainHealthCheck
    - Request variables
    - Response variables
    - Request example with active providers
    - Response example with active providers
    - Request example with no providers active
    - Response example with no providers active
  - b) GlobalHealthCheck
    - Request variables
    - Response variables
    - Request example
    - Response example
- 9. Linked Players Methods



- GetUserLinkedPlayers
- Deprecated Methods
- Bonus
  - a) ApplyUserBonus(DEPRECATED)
    - Request variables
    - Response variables
    - Request example
    - Request example with lovation enabled
    - Response example
  - b) GetUserBonusDetails(DEPRECATED)
    - Response variables
    - Response example
  - c) ForfeitUserCasinoBonus(DEPRECATED)
    - Request variables
    - Request example
    - Response example
  - d) ApplyAffiliateChipBonus(DEPRECATED)
    - Request variables
    - Response variables
    - Request example
    - Response example
  - e) ApplyUserExternalBonus(DEPRECATED)
    - Request variables (only for POST methods)
    - Response variables
    - Request example
    - Request example with lovation enabled
    - Response example
  - f) GetAvailableBonusList(DEPRECATED)
    - Request variables
    - Response variables
    - Response example
  - g) ApplicableBonusList(DEPRECATED)
    - Request variables
    - Response variables
    - Request example
    - Response example
  - h) SetUserPromotion(DEPRECATED)
    - Request variables
    - Response variables
    - Request example
    - Response example
  - i) GetUserPromotions(DEPRECATED)
    - Request variables
    - Response variables
    - Request example
    - Response example
- Transactions
  - a) ManualDeposit (OBSOLETE)
    - Request variables
    - Response variables
    - Request example
    - Response example
  - b) ManualWithdraw (OBSOLETE)
    - Request variables
    - Response variables
    - Request example
    - Response example
- Appendix 1. Error codes
- Appendix 2. Currency codes
- Appendix 3. Country codes
- Appendix 4. Phone country codes

- Appendix 5. Language codes

## Environments URLs

Development: <http://core2.gm.dev.everymatrix.com/serverapi>

Stage: <https://core-gm-stage.everymatrix.com/serverapi>

Production:

- <https://admin.gamatrix.com/serverapi>
- <https://admin3.gamatrix.com/serverapi>
- <https://admin4.gamatrix.com/serverapi>
- <https://admin5.gamatrix.com/serverapi>

## Introduction

The following document contains descriptions concerning operators' integration into GamMatrix and GamMatrix Server API services. From here on, the GamMatrix back office service will be referred to as GM.

Overview GamMatrix Server API is a server-to-server REST based protocol using JSON format. Data encryption, confidentiality and security are provided by using HTTPS.

To every operator a PartnerID and PartnerKey is given, which must be supplied with every API call. GM restricts API access by the operator's provided IP address(es).

See the list of API methods below according to the table structure:

1. Name - parameter variable name
2. M/O - Mandatory / Optional flag which indicates whether the parameter is mandatory or optional
3. Type - data type
4. Size - size restrictions
5. Comments - general notes about the parameter

## 1) Pre-Registration

### a) IsUserNameAvailable

Check if username is available in the system.

URL: [|GmServiceURL|/ServerAPI/IsUserNameAvailable/{VERSION}/{PARTNERID}/{PARTNERKEY}/{USERNAME}](#)

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isAvailable	int	1	1 - available, 0 - not available
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  }
}
```

```

},

"isAvailable": 0,

"success": 1,

"timestamp": "2018-08-08 14:51:22",

"version": "1.0"

}

```

## b) IsEmailAvailable

Check if email address is available in the system.

URL: |GmServiceURL|/ServerAPI/IsEmailAvailable/{VERSION}/{PARTNERID}/{PARTNERKEY}/{EMAIL}

HTTP Method: GET

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isAvailable	int	1	1 - available, 0 - not available
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Response example

```

{

  "errorData": {

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

  },

  "isAvailable": 0,

  "success": 1,

  "timestamp": "2018-08-08 14:49:18",

  "version": "1.0"

}

```

## c) IsAliasAvailable

Check if alias is available in the system.

URL: |GmServiceURL|/ServerAPI/IsAliasAvailable/{VERSION}/{PARTNERID}/{PARTNERKEY}/{ALIAS}

HTTP Method: GET

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)

*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isAvailable	int	1	1 - available, 0 - not available
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isAvailable": 1,
  "success": 1,
  "timestamp": "2018-08-08 14:47:24",
  "version": "1.0"
}
```

#### d) CheckUserExists

Check if user exists in the system. Returns the UserId and user's account status.  
 URL: |GmServiceURL|/ServerAPI/CheckUserExists/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
birthDate	O	String		Date of birth. Format yyyy-mm-dd
email	O	String	5-50	Email address
firstName	O	string	2-50	The player's first name
lastName	O	string	2-50	The player's last name
phone	O	string	7-30	The player's phone number
zip	O	string	20	Zip/Postal code
personalId	O	string	30	The player's personal ID.

#### Response variables

Name	Data type	Size	Comments
activeStatus	int		Inactive = -1, Active = 0, Blocked = 1, Closed = 2
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system

success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
userId	long		User ID
version	string		Version used, set to "1.0"

#### Request example

```
{
  "birthDate": "1986-04-24",
  "email": "582fbe25464c46a29cda26805ab776ab@test.com",
  "firstName": "8eff1e5752c845feaa31bc77eae25f46",
  "lastName": "b2fd3f608de448e1892731bdf8f4b95b",
  "phone": "26383514",
  "zip": "XX"
  "personalId": "1860424231554"
}
```

#### Response example (success)

```
{
  "activeStatus": 0,
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-06-28 12:45:54",
  "userId": 2005458,
  "version": "1.0"
}
```

#### Response example (user does not exist)

```
{
  "activeStatus": 0,
  "errorData": {
    "errorCode": 1000,
    "errorDetails": [],
    "errorMessage": "User not found (User does not exist)",
    "logId": 51504464
  },
  "success": 0,
  "timestamp": "2018-08-08 08:14:33",
  "userId": 0,
  "version": "1.0"
}
```

#### Response example (empty request)

```
{
  "activeStatus": 0,
```

```

"errorData": {
    "errorCode": 1000,
    "errorDetails": [],
    "errorMessage": "No selection criteria defined",
    "logId": 179917
},
"success": 0,
"timestamp": "2018-06-28 12:54:17",
"userId": 0,
"version": "1.0"
}

```

## 2. Registration

### a) RegisterUser

Register a new user in the system.

URL: |GmServiceURL|/ServerAPI/RegisterUser/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M / O	Data type	Size	Comments
activateAccount	O	string	1	Flag indicating whether to automatically activate user account upon registration without sending the activation email. Set "1" for yes, otherwise "0". Default value is "0"
address1	M	string	100	Address1
address2	O	string	100	Address2
alias	O	string		Alias. Default value is username
allowNewsAndOffers	O	string	1	Flag, indicating whether to allow sending news and offers to user via email. Set "1" for yes, otherwise "0". Default value is 0.
allowSMSNewsAndOffers	O	string	1	Flag, indicating whether to allow sending news and offers to user via SMS. Set "1" for yes, otherwise "0". Default value is 0.
birthDate	M	string		Date of birth. Format yyyy-mm-dd
btag	O	string	64	Affiliate tracking code
city	M	string	50	City
countryCode	M	string	2	Country code. See Appendix 3. Country codes
currency	M	string	3	User account currency. See Appendix 2. Currency codes
email	M	string	5-50	Email address
firstName	M	string	2-50	First name
gender	O	string	1	Gender. Pass "M" for mail and "F" for female. Other values set gender to unknown.
language	O	string	2	User preferred language. See Appendix 5. Language codes
lastName	M	string	2-50	Last name
mobile	M	string	5-30	Mobile phone number
mobilePrefix	M	string	5	Mobile phone country code prefix. See Appendix 4. Phone country codes
password	M	string	8-20	Plain text password
phone	O	string	5-30	Phone number
phonePrefix	O	string	5	Phone country code prefix. See Appendix 4. Phone country codes

postalCode	M	string	20	Postal code
registrationChannel	O	string		Registration Channel. It needs to be one of "Desktop", "Mobile", "None". Default value: "None"
securityAnswer	O	string	120	Security answer
securityQuestion	O	string	120	Security question
signupIp	M	string	7-15	User IP (IPv4 format)
title	M	string		Title. Set to one of : "Mr.", "Mrs.", "Miss.", "Ms."
userName	M	string	4-20	Username
personalId	O	string	30	The player's personal ID.
iovationBlackBox	O	string	~2000 characters	The set of characters generated by the Iovation script embedded in the web page. This unique code corresponds to a certain device
nationality	O	string		List of countries
birthPlace	O	string		City of birth. May contain [space, full stop (.), comma (,), semi-colon (;), apostrophe ('), am-percent (&)] in between of two characters; cannot be empty
birthCountryCode	O	string		List of countries. Requests send value using GM internal country ID.

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
userId	long		User ID
version	string		Version used, set to "1.0"

### Request example

```
{
  "activateAccount": "1",
  "address1": "test1",
  "address2": "test2",
  "alias": "JDtestreg",
  "allowNewsAndOffers": "1",
  "allowSMSNewsAndOffers": "1",
  "birthDate": "1989-06-06",
  "btag": "testie",
  "city": "Rome",
  "countryCode": "IT",
  "currency": "EUR",
  "email": "testingacc@sharklasers.com",
  "firstName": "John",
  "gender": "1",
  "iovationBlackBox": "a_test_string",
```

```

"language": "EN",
"lastName": "Doyle",
"mobile": "764696969",
"mobilePrefix": "+39",
"password": "Password123",
"phone": "764696969",
"phonePrefix": "+39",
"postalCode": "969696",
"registrationChannel": "Desktop",
"securityAnswer": "brain",
"securityQuestion": "My favourite bet?",
"signuIp": "85.9.7.222",
"title": "Mr.",
"userName": "JDtestreg"
"personalId" : "1890606254897"
"birthCountryId": 12,
"nationality": "Andorran",
"birthPlace": "Caldeea"
}

```

#### Response example

```

{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-06 13:58:51",
  "userId": 3305269,
  "version": "1.0"
}

```

#### b) GetUserDetails

Returns user details.

URL: |GmServiceURL|/ServerAPI/GetUserDetails/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
------	-----------	------	----------



activeStatus	int		InActive = -1, Active = 0, Blocked = 1, Closed = 2
address1	string	100	Address1
address2	string	100	Address2
affiliateMarker	string		Affiliate tracking code
alias	string		Alias. Default value is username
allowNewsEmail	boolean	1	Flag, indicating whether to allow sending news and offers to user via email. Set "1" for yes, otherwise "0". Default value is 0.
allowSmsOffer	boolean	1	Flag, indicating whether to allow sending news and offers to user via SMS. Set "1" for yes, otherwise "0". Default value is 0.
birthDate	string		Date of birth. Format yyyy-mm-dd
city	string	50	City
countryCode	string	2	Country code. See Appendix 3. Country codes
currency	string	3	User account currency. See Appendix 2. Currency codes
email	string	5 - 50	Email address
errorData	object		If success, ErrorCode = 0, LogId = 0, ErrorDetails = "" (empty), ErrorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
firstName	string	2 - 50	First name
language	string	2	User preferred language. See Appendix 5. Language codes
lastName	string	2 - 50	Last name
mobile	string	7 - 30	Mobile phone number
mobilePrefix	string	5	Mobile phone country code prefix. See Appendix 4. Phone country codes
phone	string	7 - 30	Phone number
phonePrefix	string	5	Phone country code prefix. See Appendix 4. Phone country codes
prevLoginDate	string		Previous login date
Roles	array		List of roles assigned to this user
activeStatus	int		InActive = -1, Active = 0, Blocked = 1, Closed = 2
*authorDisplayName	string		Admin user who created the role
*description	string		Role description
*name	string		Role name
*userRoleCreated	string		Date when the role was created. Format: yyyy-MM-dd HH:mm:ss
*userRoleId	long		Unique ID for the role in the GamMatrix system
securityAnswer	string	120	Security answer
securityQuestion	string	120	Security question
signuPlp	string	7 - 15	User IP (IPv4 format)
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
title	String		Title. Set to one of : "Mr.", "Mrs.", "Miss.", "Ms."
userName	String	4 - 20	Username
version	string		Version used, set to "1.0"
zip	string	20	Zip/postal code
personalId	string	30	The player's personal ID.

nationality	string	List of countries
birthPlace	string	City of birth. May contain [space, full stop (.), comma (,), semi-colon (;), apostrophe ('), am-percent (&)] in between of two characters; cannot be empty
birthCountryCode	string	List of countries. Requests send value using GM internal country ID.

### Response example

```
{
  "activeStatus": 0,
  "address1": "test1",
  "address2": "test2",
  "affiliateMarker": "test",
  "alias": "dduat1702",
  "allowNewsEmail": true,
  "allowSmsOffer": true,
  "birthDate": "1969-06-09 00:00:00",
  "city": "dedeus",
  "countryCode": "VE",
  "currency": "EUR",
  "email": "EURtest1621@ddtest2102.com",
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "firstName": "ddtest2102",
  "language": "EN",
  "lastName": "ddtest2102",
  "mobile": "755696969",
  "mobilePrefix": "+39",
  "phone": "755696969",
  "phonePrefix": "+39",
  "prevLoginDate": "2018-08-08 14:17:06",
  "roles": [
    {
      "activeStatus": 0,
      "authorDisplayName": "ddtest2102 ddtest2102",
      "description": "Trusted user",
      "name": "Trusted User",
      "userRoleCreated": "2018-08-06 14:07:04",
      "userRoleId": 3634790
    }
  ]
}
```

```

    },
    {
      "activeStatus": 0,
      "authorDisplayName": "ddtest2102 ddtest2102",
      "description": "User identity is verified",
      "name": "Verified Identity",
      "userRoleCreated": "2017-04-20 11:13:34",
      "userRoleId": 3583327
    }
  ],
  "securityAnswer": "brain",
  "securityQuestion": "My favourite bet?",
  "signupIp": "127.0.0.1",
  "success": 1,
  "timestamp": "2018-08-08 14:30:37",
  "title": "Mr.",
  "userName": "EURtest1621",
  "version": "1.0",
  "zip": "969696"
  "personalId" : "1690609414827"
  "birthCountryId":12,
  "nationality": "Andorran",
  "birthPlace": "Caldeea"
}

```


### c) UpdateUserDetails

Updates user registration details.

URL: |GmServiceURL|/ServerAPI/UpdateUserDetails/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M /O	Data type	Size	Comments
address1	M	string	100	Address1
address2	O	string	100	Address2
affiliateMarker	O	string		Marks the player under a specific affiliate if needed. <div>  <b>Note - If btag parameter is null during registration process when received then this parameter is ignored and no update is performed.</b> </div>
alias	O	string		Alias. Default value is username
allowNewsEmail	O	boolean		Flag, indicating whether to allow sending news and offers to user via email
allowSmsOffer	O	boolean		Flag, indicating whether to allow sending news and offers to user via SMS

birthDate	M	string		<p>Date of birth. Format yyyy-mm-dd</p> <p>The birthdate will not be updated if a valid value is already inserted in the database.</p> <p>This method will permit the overwrite, only in case of invalid birthDate from DB - DateTime.MinValue or DateTime.MaxValue.</p> <p>DateTime.MinValue: 00:00:00.0000000 UTC, January 1, 0001  DateTime.MaxValue 23:59:59.9999999 UTC, December 31, 9999</p>
gender	O	string		Gender of the user
city	M	string	50	City
email	M	string	5 - 50	Email address
firstName	M	string	2 - 50	First name
lastName	M	string	2 - 50	Last name
mobile	M	string	7 - 30	Mobile phone number
mobilePrefix	M	string	5	Mobile phone country code prefix. See Appendix 4. Phone country codes
phone	O	string	7 - 30	Phone number
phonePrefix	O	string	5	Phone country code prefix. See Appendix 4. Phone country codes
postalCode	M	string	20	Postal code
securityAnswer	O	string	120	Security answer
securityQuestion	O	string	120	Security question
sessionId	M	string	40	User session ID. See method LoginUser for how to get SessionId
userId	O	long		The player's user ID in GM.
title	M	string		Title. Set to one of: "Mr.", "Mrs.", "Miss.", "Ms."
personalId	O	string	30	The player's personal ID.
nationality	M	string		List of countries
birthPlace	M	string		City of birth. May contain [space, full stop (.), comma (,), semi-colon (;), apostrophe ('), am-percent (&)] in between of two characters; cannot be empty
birthCountryCode	M	string		List of countries. Requests send value using GM internal country ID.

## Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

## Request example

```
{
  "address1": "test1",
  "address2": "test2",
  "affiliateMarker": "Affimarker",
  "alias": "ddtest",
  "allowNewsEmail": true,
  "allowSmsOffer": true,
  "birthDate": "1980-04-01",
  "city": "test",
  "email": "gmcoreuser@gmttests.com",
  "firstName": "dd",
  "gender": "male",
  "lastName": "test",
  "mobile": "123456789",
  "mobilePrefix": "+40",
  "phone": "1234567891",
  "phonePrefix": "+40",
  "postalCode": "5000",
  "securityAnswer": "My favourite bet?",
  "securityQuestion": "brain",
  "userId": "3030264",
  "birthCountryId": 12,
  "nationality": "Andorran",
  "birthPlace": "Caldeea"
}
```

**Response example**

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-07 13:57:34",
  "version": "1.0"
}
```

#### d) ChangeUserPassword

Changes a user's password.

URL: |GmServiceURL|/ServerAPI/ChangeUserPassword/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

##### Request variables

Name	M /O	Data type	Size	Comments
checkOldPassword	O	boolean		Flag, indicating whether to allow check old password before changing to new one. Set "true" for yes, otherwise "false". Default value is "false"
hashKey	O	string		Key used for password change
oldPlainTextPassword	M	string	8 - 20	Old password in plain text
plainTextPassword	M	string	8 - 20	New password in plain text
sessionId	M	string	40	User's session ID. See method LoginUser for how to get session ID

##### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed

timestamp	string		yyyy-MM-dd HH:mm:ss
session	string		Version used, set to "1.0"

#### Request example

```
{
  "checkOldPassword":true,
  "hashKey":"42f749ade7f9e195bf475f37a44cafcb",
  "oldPlainTextPassword":"Password123",
  "plainTextPassword":"Password1234",
  "sessionId":"3fb155db-6f11-4cd7-a53e-b23b297dff4c"
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 08:10:57",
  "version": "1.0"
}
```

### e) SetUserForgotPassInfo

SetS security question and security answer for a user. Overwrites existing information.

URL: |GmServiceURL|/ServerAPI/SetUserForgotPassInfo/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
securityAnswer	M	string	120	Security answer
securityQuestion	M	string	120	Security question
sessionId	M	string	40	User session ID. See method LoginUser for how to get SessionId

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```
{  
  
  "securityAnswer": "brain",  
  
  "securityQuestion": "My favourite bet?",  
  
  "sessionId": "255cb8f4-cc66-407e-be9a-a5bd8387ecd6"  
  
}
```

#### Response example

```
{  
  
  "errorData": {  
  
    "errorCode": 0,  
  
    "errorDetails": [],  
  
    "errorMessage": "",  
  
    "logId": 0  
  
  },  
  
  "isScheduled": 1,  
  
  "success": 1,  
  
  "timestamp": "2018-08-06 14:22:55",  
  
  "version": "1.0"  
  
}
```

#### f) GenerateUserHash

Generates the user hash.

URL: |GmServiceURL|/ServerAPI/GenerateUserHash/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
URL	M	string		Parameter received from request to get new user hash
userId	M	string		User ID

#### Response variables

Name	Data type	Size	Comments
URL			Parameter used to generate user hash
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```
{  
  
  "URL": "https://megavinn.stage.everymatrix.com/example.com",  
  
}
```



```
"UserId": "2977230"
```

```
}
```

#### Response example

```
{
  "URL": "19668c087e804656b321b4af22e4b7a8",
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-10 09:09:26",
  "version": "1.0"
}
```

#### g) IsUserHashValid

Checks if the user hash is valid.

URL: |GmServiceURL|/ServerAPI/IsUserHashValid/{VERSION}/{PARTNERID}/{PARTNERKEY}/{HASHKEY}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isValid	int		Check if user hash is not null and user hash status is active. If success - 1, failed - 0
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isValid": 0,
  "success": 1,
  "timestamp": "2018-08-08 14:50:05",
  "version": "1.0"
}
```

#### h) ActivatePlayer

Activates player in the system once he/she clicks activation link.

URL: |GmServiceURL|/ActivatePlayer/{VERSION}/{PARTNERID}/{PARTNERKEY}/{HASHKEY}

HTTP Method: GET

## Request variables

Name	M/O	Type	Size	Comments
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
hashKey	M	string		Unique hash value, obtained by calling <i>GenerateUserHash</i> method

## Response variables

Name	M/O	Type	Size	Comments
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
sessionId		int	1	User session identifier (i.e. user is considered as logged-in)
userId		long	1	User identifier (i.e. user is considered as logged-in)
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

## Request example

http://localhost/ServerAPI/ActivatePlayer/1.0/testID/testCode/

## Response example Successful response

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "sessionId": "ae846f35-8ed0-40d3-a50f-6f31fff2b9c2",
  "success": 1,
  "timestamp": "2016-07-06 09:32:54",
  "userId": 112912,
  "version": "1.0"
}
```

## Response example Failed response (user is already activated)

```
{
  "errorData": {
    "errorCode": 1027,
```

```

        "errorDetails": [],
        "errorMessage": "User account is already active",
        "logId": 735593
    },
    "sessionId": null,
    "success": 0,
    "timestamp": "2016-07-06 09:34:08",
    "userId": 0,
    "version": "1.0"
}

```

#### Response example Failed response (invalid or expired hash)

```

{
    "errorData": {
        "errorCode": 1028,
        "errorDetails": [],
        "errorMessage": "Invalid or expired hash",
        "logId": 735595
    },
    "sessionId": null,
    "success": 0,
    "timestamp": "2016-07-06 09:34:40",
    "userId": 0,
    "version": "1.0"
}

```

## 3. Login

### a) LoginUser

Logs in a user to the system.

URL: |GmServiceURL|/ServerAPI/LoginUser/ {VERSION}/{PARTNERID}/{PARTNERKEY}/{USERNAME}/{PASSWORD}/{IP}

or:

URL: |GmServiceURL|/ServerAPI/LoginUser/ {VERSION}/{PARTNERID}/{PARTNERKEY}/{EMAIL}/{PASSWORD}/{IP}


or:

URL:|GmServiceURL|/ServerAPI/LoginUser/ {VERSION}/{PARTNERID}/{PARTNERKEY}/{USERNAME}/{TOKEN}/{IP}

HTTP Method: GET

#### Request variables

Name	M /O	Data type	Size	Comments
version	M	String		Version used, set to "1.0"
partnerId	M	String		partnerId, provided by integration manager.

partnerKey	M	String		partnerKey, provided by integration manager.
username or email	M	String		Valid user name or email address.
password	M	String		Valid user password, always to be sent hashed in MD5. Can also be plain text for initial phases of the integration.
token		String		Pre-established token, used to authenticate the user without his personal password.  <div>  <b>To use token</b>  Username/Token for login. Feature can be enabled on demand, the token needs to be established and set up in EveryMatrix BackEnd. </div>
ip	M	String		Valid IPv4 user ip address.

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
firstName	string	2 - 50	First name
language	string	2	User preferred language
lastName	string	2 - 50	Last name
sessionId	string	36, reserve 64 (GUID)	User's session ID. The session id is kept on the GamMatrix back end and is transmitted to other modules/third parties
registrationDate	date		Returns the registration timestamp of the player in Unix format.
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
userId	long		User ID
version	string		Version used, set to "1.0"

#### Request examples

- **Username:**

<https://core-gm-stage.everymatrix.come/ServerAPI/LoginUser/1.0/megaid1008/megaCode1008/EURtest1621/Password123/127.0.0.1>

- **Email:**

<https://core-gm-stage.everymatrix.come/ServerAPI/LoginUser/1.0/megaid1008/megaCode1008/EURtest1621@sharklasers.com/Password123/127.0.0.1>

- **Username/Token:**

<https://core-gm-stage.everymatrix.come/ServerAPI/LoginUser/1.0/megaid1008/megaCode1008/EURtest1621/TOKEN/127.0.0.1>

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "firstName": "test",
  "language": "EN",
  "lastName": "user",
  "registrationDate": "/Date(1549287669963+0000)/",
  "sessionId": "0bf465bc-cc8d-4b85-80bc-6eafa1103dd2",
  "success": 1,
  "timestamp": "2019-02-12 15:12:06",
  "userId": 3328603,
  "version": "1.0"
}
```

#### b) LoginUser (with lovation enabled)

Logs in a user in the system.

URL: |GmServiceURL|/ServerAPI/LoginUser/ {VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M /O	Data type	Size	Comments
userName	M	string		Username
password	M	string		User's password
ip	M	string		User's IP (IPv4 format)
iovationBlack Box	O	string	~2000 characters	The set of characters generated by the lovation script embedded in the web page. This unique code corresponds to a certain device

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Error details
*errorMessage	string		Error message
*logId	int		Error log ID in the system
success	int	1	1 – success, 0 – failed
timestamp	string		yyyy-MM-dd HH:mm:ss
firstName	String		User's first name
language	string		User's preferred language
lastName	string		User's last name
sessionId	string		User's session Id. The session id is kept on the GamMatrix backend and is transmitted to other modules/ third parties.
userId	string		User ID
version	string		Version used, set to 1.0.

#### Request example (only for POST methods)

```
{
  "iovationBlackBox": "QUMXGB0poNf94lis1ztu1p0Y9k4RDY0Hfp5ie720zxMSrvHBQm916",
  "ip": "127.0.0.1",
  "password": "Password123",
  "userName": "EURtest1621"
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "firstName": "ddtest2102",
  "language": "EN",
  "lastName": "ddtest2102",
  "sessionId": "383034cd-093e-4e1b-836b-e8e51c32b1ef",
  "success": 1,
  "timestamp": "2018-08-06 11:53:51",
  "userId": 3201254,
  "version": "1.0"
}
```

#### c) LogoutUser

Logs a user out of the system.

URL: |GmServiceURL|/ServerAPI/LogoutUser/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-06 13:31:07",
  "version": "1.0"
}
```

#### d) IsActiveUserSession

Checks if user has an active session.

URL: |GmServiceURL|/ServerAPI/IsActiveUserSession/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
firstName	string	2 - 50	First name. Returned if IsActive = 1
isActive	int	1	1 - active, 0 - not active
language	string	2	User preferred language. Returned if IsActive = 1. See Appendix 5. Language codes
lastName	string	2 - 50	Last name. Returned if IsActive = 1
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss

userId	long		User ID. Returned if IsActive = 1
userName	string	4 - 20	Username. Returned if IsActive = 1
version	string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "firstName": "ddtest2102",
  "isActive": 1,
  "language": "EN",
  "lastName": "ddtest2102",
  "success": 1,
  "timestamp": "2018-08-08 14:46:42",
  "userId": 3201254,
  "userIp": "127.0.0.1",
  "userName": "EURtest1621",
  "version": "1.0"
}
```

#### e) IsValidCredentials

Checks if the user inserted valid credentials.

URL:|GmServiceURL|/ServerAPI/IsValidCredentials/{VERSION}/{PARTNERID}/{PARTNERKEY}/{USERNAME}/{PASSWORD}  
HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isValid	int		Check if username and password are valid. If succes - 1, failed - 0
success	int	1	1 - success, 0 - failed
timeStamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
```



```

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

},

"isValid": 1,

"success": 1,

"timestamp": "2018-08-08 14:51:56",

"version": "1.0"

}

```

---

## 4. User account

### a) GetUserAccounts

Returns a list of all gaming accounts for a user. This is used for obtaining the user `currency` and `walletId`, to be used in Payment processing methods (Deposit, Withdraw)

URL: |GmServiceURL|/ServerAPI/GetUserAccounts/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
accounts	array		Returns a list of gaming accounts for a user
*balanceAmount	string		Gaming account real money balance. A maximum of 2 decimal places can be used after the integer for proper functionality.
*bonusAmount	string		Bonus balance in the gaming account. A maximum of 2 decimals places can be used after the integer for proper functionality.
*currency	string		User account currency
*displayName	string		Gaming account display name
*id	long		Gaming account id in the GamMatrix system
*isBalanceAvailable	string	1	"0" - false, "1" - true
*omBonusAmount	string		OddsMatrix account bonus balance. A maximum of 2 decimals places can be used after the integer for proper functionality.
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isNegativeBalance	string		Check if user account has negative balance. If true - 1, if false -0
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```

{

  "accounts": [

    {

      "balanceAmount": "0.00",

      "bonusAmount": "0.00",

      "currency": null,

```

```
"displayName": "Casino",
"id": 8489991,
"isBalanceAvailable": "0",
"omBonusAmount": "0.00"
},
{
  "balanceAmount": "9893.42",
  "bonusAmount": "0.00",
  "currency": "EUR",
  "displayName": "Casino",
  "id": 8363737,
  "isBalanceAvailable": "1",
  "omBonusAmount": "0.00"
},
{
  "balanceAmount": "0.00",
  "bonusAmount": "0.00",
  "currency": "EUR",
  "displayName": "Casino",
  "id": 9490287,
  "isBalanceAvailable": "1",
  "omBonusAmount": "0.00"
},
{
  "balanceAmount": "0.00",
  "bonusAmount": "0.00",
  "currency": null,
  "displayName": "TransferWallet",
  "id": 9490288,
  "isBalanceAvailable": "0",
  "omBonusAmount": "0.00"
},
{
  "balanceAmount": "10.00",
  "bonusAmount": "0.00",
  "currency": "EUR",
  "displayName": "UBS",
  "id": 8363738,
```

```

        "isBalanceAvailable": "1",
        "omBonusAmount": "0.00"
    }
},
"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
"isNegativeBalance": "0",
"success": 1,
"timestamp": "2018-08-02 14:04:35",
"version": "1.0"
}

```

## b) GetUsersRoles

Returns a list of all available roles in the system.

URL: |GmServiceURL|/ServerAPI/GetUsersRoles/ {VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: GET

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, <ul style="list-style-type: none"> <li>• errorCode = 0,</li> <li>• logId = 0,</li> <li>• errorDetails = "" (empty),</li> <li>• errorMessage = "" (empty)</li> </ul>
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
usersRolesList	array		Roles available in the system that can be assigned to players
*description	string		Role description
*id	long		Internal role ID in the GamMatrix system
*name	string		Role name
version	string		Version used, set to "1.0"

### Response example

```

{
    "errorData":{
        "errorCode":0,
        "errorDetails": "",

```

```

        "errorMessage": "",
        "logId": 0
    },
    "success": 1,
    "timestamp": "2018-08-21 11:04:35",
    "usersRolesList": [{
        "description": "Test role description",
        "id": 20368,
        "name": "Test Role"
    }],
    "version": "1.0"
}

```

### c) AssignUserRole

Assigns roles to a user. See GetUsersRoles method to get a list of all available roles in the system. Only one role can be assigned with one request.  
 URL: |GmServiceURL|/ServerAPI/ AssignUserRole/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
roleId	M	long		ID of the role to be assigned. See method GetUsersRoles for complete list of available roles
userId	M	string	40	User ID

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```

{
    "roleId": 16716,
    "userId": "3201254"
}

```

#### Response example

```

{
    "errorData": {
        "errorCode": 0,
        "errorDetails": [],

```

```

    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 08:07:02",
  "version": "1.0"
}

```

#### d) AssignUserRoleWithExpiration

Assigns roles to a user with an expiration date/period. See GetUsersRoles method to get a list of all available roles in the system. Only one role can be assigned per request.

URL: |GmServiceURL|/ServerAPI /AssignUserRoleWithExpiration/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M / O	Data type	Size	Comments
expirationDate	O	string		The date at which the role will expire
expirationPeriodNumber	O	int		The number of days/weeks/months after which the role will expire
expirationPeriodType	O	object		The period type selected to measure the expiration period: 1, 2, 3. The parameter "1" is for days, "2" is for weeks and "3" for months, when an expiration period is selected
roleId	M	long		Id of the role to be assigned. See method GetUsersRoles for complete list of available roles
userId	M	string		User ID

Only one of the "ExpirationDate" and "ExpirationPeriodNumber" parameters can return valid values at a time.

"ExpirationPeriodNumber" cannot return valid values without "ExpirationPeriodType".

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

If an expiration date is selected

```

{
  "expirationDate":"2019-09-16",
  "expirationPeriodNumber":2147483647,
  "expirationPeriodType":0,
  "roleId":16716,
  "userID":"3201254 "
}

```

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 08:09:32",
  "version": "1.0"
}
```

### e) RemoveUserRole

Removes assigned roles from a user.

URL: |GmServiceURL|/ServerAPI/RemoveUserRole/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
roleId	M	long		ID of the role to be removed from the user.
userId	M	string		User ID

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Request example

```
{
  "roleId":3583327,
  "userId":"3201254"
}
```

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
```

```

    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-06 14:10:56",
  "version": "1.0"
}

```

#### f) SetUserMetadata

Sets a metadata key/value on a user. GM Core metadata is a simple key/value system for attaching generic information to a user's record within the GM Core database.

URL: |GmServiceURL|/ServerAPI/SetUserMetadata/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
metadataKey	M	string		The metadata key
metadataValue	O	string		The metadata value. If empty, metadata key is removed from system.
sessionId	M	string		User's session ID

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```

{
  "metadataKey": "PM2AGENTID",
  "metadataValue": "3196922 ",
  "sessionId": "255cb8f4-cc66-407e-be9a-a5bd8387ecd6"
}

```

#### Response example

```

{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,

```

```
"timestamp": "2018-08-06 14:33:40",  
"version": "1.0"  
}
```

### g) GetUserMetadata

Retrieves the metadata value associated with a particular user & key.

URL: |GmServiceURL|/ServerAPI/GetUserMetadata/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{METADATAKEY}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
metadataValue	string		The user's parent ID
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```
{  
  "errorData": {  
    "errorCode": 0,  
    "errorDetails": [],  
    "errorMessage": "",  
    "logId": 0  
  },  
  "metadataValue": null,  
  "success": 1,  
  "timestamp": "2018-08-08 14:32:08",  
  "version": "1.0"  
}
```

### h) GetUserAssignedRoles

This method returns the roles assigned to for the given userid

URL: |GmServiceURL|/ServerAPI/GetUserAssignedRoles/{VERSION}/{PARTNERID}/{PARTNERKEY}/{USERID}

HTTP Method: GET

#### Request variables

Name	M/O	Type	Size	Comments
userid	M	long		The player's user ID.

#### Response variables

Name	M/O	Type	Size	Comments
------	-----	------	------	----------



assignedUserRolesList		object	Object describing the role details
*description		string	Description of the role
* name		string	Name of the role
* roleId		long	Role id in the system
* userRoleExpirationText		string	Information about the role expiry date
errorData		object	If success, <ul style="list-style-type: none"> <li>• errorCode = 0,</li> <li>• logId = 0,</li> <li>• errorDetails = "" (empty),</li> <li>• errorMessage = "" (empty)</li> </ul>
*errorCode		int	Error code
*errorDetails		string	Possible error details
*errorMessage		string	Error message
*logId		int	Error log ID in the system
Timestamp	M	datetime	Date and time of the request
Version	M	string	Version used, set to "1.0"

### Response example

```
{
  "assignedUserRolesList": [
    {
      "description": "User identity is verified",
      "name": "Verified Identity",
      "roleId": 522,
      "userRoleExpirationText": "No Expiration"
    }
  ],
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-02 14:10:44",
  "version": "1.0"
}
```

### i)SearchUsers

This method allows the back-office user to identify players based on a given partial string by searching the player base either by personal ID or username.

URL: |GmServiceURL|/ServerAPI/SearchUsers/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

### Request variables

Name	M/O	Type	Size	Comments
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
searchString	M	string		Search string value.
byPersonalId	O	bool		Possible values: 'true' or 'false' <b>True</b> , if searching by personalID.
byUserName	O	bool		<b>True</b> , if searching by username.
pageNumber	M	int		Page number (starting from 0)
pageSize	M	int		Page size (1-500)

#### Response variables

Name	Type	Size	Comments
success	int	1	Can be either: <ul style="list-style-type: none"> <li>1 – success</li> <li>0 – failed</li> </ul>
timestamp	string		The date and time, formatted as follows: <b>yyyy-MM-dd HH:mm:ss</b> .
version	string		Version used, set to "1.0"
errorData	object		If success, <ul style="list-style-type: none"> <li>errorCode = 0</li> <li>logId = 0</li> <li>errorDetails=[] (empty)</li> <li>errorMessage="" (empty)</li> </ul>
• errorCode	int		Error code
• errorDetails	string		Possible error details
• errorMessage	string		Error message
• logId	long		Error log ID in the system
totalRecords	int		Total result records number
users	object		Search result user list
• userId	long		Player's identification.
• userName	string		Player's username
• countryCode	string		Player's country code
• activeStatus	Int		Player's active status: <ul style="list-style-type: none"> <li>InActive = -1</li> <li>Active = 0</li> <li>Blocked = 1</li> <li>Closed = 2</li> </ul>

#### Request Example

##### By personal id

http://localhost/ServerAPI/SearchUsers/1.0/JetId/JetCode

```
{
  "searchString": "xxx123",
```

```
"byPersonalId":true,
"pageNumber":0,
"pageSize":50
}
```

#### By username

http://localhost/ServerAPI/SearchUsers/1.0/JetId/JetCode

```
{
  "searchString":"ab1e5576cc334e3",
  "byUserName":true,
  "pageNumber":0,
  "pageSize":50
}
```

#### Response Example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-04-30 14:01:48",
  "totalRecords": 1,
  "users": [
    {
      "activeStatus": 0,
      "countryCode": "RO",
      "userId": 2005458,
      "userName": "test123"
    }
  ],
  "version": "1.0"
}
```

#### j) SetUserStatus

Gives the ability to change a player's blocked status.

URL: |GmServiceURL|/SetUserStatus/{version}/{partnerID}/{partnerKey}

HTTP Method: POST

#### Request Variables

Name	Data Type	Size	Comments
userId	string		The player's unique user ID in GamMatrix back-end
newStatus	enum		The new status about to be applied to the player account. Possible values are: <ul style="list-style-type: none"> <li>0 - Active</li> <li>1 - Blocked</li> </ul>
blockType	enum		The reason for which the account has been blocked. Possible values are: <ul style="list-style-type: none"> <li>1 - ChargeBack</li> <li>2 - PokerChipDumping</li> <li>3 - UserRequest</li> <li>4 - BonusAbuse</li> <li>5 - PendingInvestigation</li> <li>6 - SuspectFraud</li> <li>7 - ConfirmedFraud</li> <li>8 - PromotionAbuse</li> <li>10 - Other</li> <li>11 - DuplicateAccount</li> <li>12 - TCDeclined</li> <li>13 - GamblingProblem</li> <li>15 - IncorrectPersonalData</li> </ul>
blockNote	string		Notes regarding the block action. Can be viewed in Audit.

### Request Example

<https://admin3.gammatrix.com/ServerAPI/SetUserStatus/1.0/jetID/jetCode/>

```
{
  "UserID":1234567,
  "NewStatus":1,
  "BlockType":2,
  "BlockNote":"test block"
}
```

### Response Variables

Name	Data type	Size	Comments
errorData	<i>object</i>		If success: <ul style="list-style-type: none"> <li>errorCode = 0</li> <li>logId = 0</li> <li>errorDetails = "" (empty)</li> <li>errorMessage = "" (empty)</li> </ul>
*errorCode	<i>int</i>		The error code.
*errorDetails	<i>string</i>		Possible error details.
*errorMessage	<i>string</i>		The error message.

*logId	long		The error log identification number in the system.
success	int	1	Possible values are: <ul style="list-style-type: none"> <li>1 - success</li> <li>0 - failed.</li> </ul>
timestamp	string		The date and time, formatted as follows: <b>yyyy-MM-dd HH:mm:ss</b> .
version	string		Version used, set to "1.0".

## Response Example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2019-11-21 12:37:44",
  "version": "1.0"
}
```

## 5. Payment

### a) RegisterPayCard

Registers user's paycard. Only for Credit Cards.

URL: |GmServiceURL|/RegisterPayCard/{version}/{partnerID}/{partnerKey}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
brandType	M	string	2	VI for Visa, VE for Visa Electron, VD for Visa Delta, MC for Master Card, MA for Maestro International, MD for Master Credit Debit SW for Switch, SO for Solo
displayName	O	string		Optional parameter used to display the masked Credit Card Number in the Admin interface. For credit cards registration DisplayName is automatically formatted by GM Core (e.g. 675595..4868)
displayName	O	string		Optional parameter used to display the masked Credit Card Number in the Admin interface. For credit cards registration DisplayName is automatically formatted by GM Core (e.g. 698554..5454)
expiryDate	M	string		yyyy-mm
identityNumber	M	string		Credit card number
issueNumber	O	string		Credit card issue number. Only for MasterCard Maestro and Switch card types
issuerCompany	O	string		Credit card issuing company
issuerCountry	M	string	2	Credit card issuer country code. See Appendix 3. Country codes
ownerName	M	string		Card holder name
sessionId	M	string	40	User's session ID. See method LoginUser for how to get SessionId
skipValidation	O	boolean		When set to true, BIN validation is skipped

vendorId	M	int		Always set to 1011 (Payment Trust Vendor)
----------	---	-----	--	---

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
payCardId	long		Unique ID of the created PayCard
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Request example

```
{
  "sessionId":"77c6c460-d8c6-48b7-a1be-9aee7b7e0e02",
  "vendorId":"1011",
  "identityNumber":"4929413944814001",
  "issueNumber":"548",
  "displayNumber":"492942...001",
  "displayName":"492942...001",
  "skipValidation":"true",
  "ownerName":"Dd Rr",
  "issuerCountry":"EUR",
  "expiryDate":"2019-09-01",
  "brandType":"VI"
}
```

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "payCardId": 2167929,
  "success": 1,
  "timestamp": "2018-08-08 14:21:26",
  "version": "1.0"
}
```

### b) GetUserPayCards

Returns a list of user's registered Paycards.

URL: |GmServiceURL|/ServerAPI/GetUserPayCards/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
payCards	array		List of registered Paycards for the user
*activeStatus	int		InActive = -1, Active = 0, Blocked = 1, Closed = 2
*displayName	string		For credit cards DisplayName is automatically formatted by GM Core (e.g. 586985..4785)
*displayNumber	string		For credit cards, DisplayNumber is automatically formatted by GM Core (e.g. 586985..4785)
*expiryDate	string		Credit card expiration date. Format: yyyy-mm
*payCardId	long		Unique ID for the Paycard in the GamMatrix system
*registered	string		Date and time of the Paycard registration. Format: yyyy-MM-dd HH:mm:ss
*successDepositBookedAmount	decimal		Sum of successful deposits with the Paycard
*successDepositNumber	long		Number of successful deposits with the PayCard
*successWithdrawBookedAmount	decimal		Sum of successful withdrawals with the Paycard
*successWithdrawNumber	long		Number of successful withdrawals with the PayCard
*userDisplayName	string		User display name
*vendorDisplayName	string		Vendor display name
*vendorName	string		Vendor name
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, Set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "payCards": [
    {
      "activeStatus": 0,
      "displayName": "422942...039",
      "displayNumber": "422941...4039",
      "expiryDate": "2019-09-30 23:59:59",
      "payCardId": 2167929,
      "registered": "2018-08-08 14:17:28",
```

```

    "successDepositBookedAmount": 0,

    "successDepositNumber": 0,

    "successWithdrawBookedAmount": 0,

    "successWithdrawNumber": 0,

    "userDisplayName": "ddtest2102 ddtest2102",

    "vendorDisplayName": "Credit Card",

    "vendorName": "PaymentTrust"

  }

],

"success": 1,

"timestamp": "2018-08-08 14:19:14",

"version": "1.0"

}

```

### c) Deposit

Makes a Credit Card deposit. This method performs the call and necessary verifications for deposit transactions (amount, currency, vendor, transaction reference, etc.).

URL: |GmServiceURL|/ServerAPI/Deposit/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Type	Size	Comments
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	M/O	string		User session identifier. Mandatory if <i>userId</i> is not provided.
userId	O/M	string		The user unique id in GM. Mandatory if <i>sessionId</i> is not provided.
debitPayCardId	O	long		User's debit paycard identifier. If not provided, <i>debitPayCardVendorId</i> should be passed
debitPayCardVendorId	O	int		User's debit paycard vendor identifier. If not provided, <i>debitPayCardId</i> should be passed. If not provided, <i>ExternalCashier vendorId</i> is used by default ( <i>debitPayCardVendorId</i> =1063).
creditAccountId	M	long		User's credit account identifier. Received from <i>GetUserAccounts</i> API. If not provided, the player's main real money wallet is used by default.
requestAmount	M	decimal		Requested deposit amount. A maximum of 2 decimals must be used after the integer for proper functionality.
requestCurrency	M	string		Requested deposit amount's currency
secretKey	O	string		Pay card secret code (specific for certain deposit methods)
transactionReference	M	string		Operator's transaction reference
note	O	string		User note
userIp	M	string		User IP (IPv4 format)
postBackURL	O	string		Postback URL (specific for certain deposit methods)
cancelURL	O	string		Cancel URL (specific for certain deposit methods)
returnURL	O	string		Return URL (specific for certain deposit methods)
applyBonusVendorID	O	int		Bonus vendor identifier. If not provided, user's bonus wallet will be used by default.
applyBonusCode	O	string		Bonus code
isMobile	O	int		"1" if it is mobile, otherwise "0"



depositSource	O	int		Source of the tx: NotAvailable=0, Unknown=1, MainDepositPage=2, QuickDepositPage=3
iovationBlackBox	O	string		Device unique code
paymentMethod	O	string		Name of the payment method, used for only Trustly vendor. Value is "Trustly. PayNPlay" by default.

#### Response variables

Name	M/O	Type	Size	Comments
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
transactionId		string		GMCore transaction identifier
balanceAmount		decimal		User's wallet balance amount after transaction (if available, returned in <i>RequestCurrency</i> ). A maximum of 2 decimals must be used after the integer for proper functionality.
transStatus		int		GMCore transaction status.  Success = 1,  Failed = 2,  Pending = 4,  DebitFailed = 6,  CreditFailed = 8,  Cancelled = 9,  RollBack = 10,  PendingNotification = 11,  PendingApproval = 12
redirectForm		string		Redirect form (specific for certain deposit methods)
redirectURL		string		Redirect URL (specific for certain deposit methods)
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

http://localhost/ServerAPI/Deposit/1.0/testID/testCode

```
{
  "sessionId": "87a2ab7a-f32f-43df-8a6e-6ed33eb39a1b",
  "userId": "320154",
  "debitPayCardId": 0,
  "debitPayCardVendorId": 1063,
  "creditAccountId": 1524784,
  "requestAmount": 5,
  "requestCurrency": "EUR",
  "transactionReference": "e718eab397054b6a8c782ed9c4a88477",
```

```

    "userIp": "127.0.0.1",

    "note": "Test deposit note",

    "applyBonusVendorID": 0,

    "applyBonusCode": null,

    "depositSource": 0,

    "iovationBlackBox": null

    "paymentMethod": "Trustly.PayNPlay"
}

```

#### Response example

```

{
    "balanceAmount": "39.24",

    "errorData": {
        "errorCode": 0,

        "errorDetails": [],

        "errorMessage": "",

        "logId": 0
    },

    "redirectForm": null,

    "redirectURL": null,

    "success": 1,

    "timestamp": "2016-06-27 13:37:16",

    "transStatus": 1,

    "transactionId": "1680118",

    "version": "1.0"
}

```

#### d) Withdraw

Makes a withdrawal. This method performs the call and necessary verifications for withdraw transactions (amount, currency, vendor, transaction reference, etc.).

URL: |GmServiceURL|/ServerAPI/Withdraw/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Type	Size	Comments
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	M/O	string		User session identifier. Parameter is mandatory if <i>userId</i> is not provided.
userId	O /M	string		The unique user id in GM. Parameter is mandatory if <i>sessionId</i> is not provided.
debitAccountID	M	long		User's debit account identifier. Received from <i>GetUserAccounts</i> API. If not provided, player's main real money wallet is used.
creditPayCardId	O	long		User's credit paycard identifier. If not provided, <i>creditPayCardVendorId</i> should be passed

creditPayCardVendorId	O	int		User's credit paycard vendor identifier. If not provided, <i>the last paycard used to deposit which also supports withdrawals is used.</i>
requestAmount	M	decimal		Requested withdraw amount. A maximum of 2 decimals places can be used after the integer for proper functionality.
requestCurrency	M	string		Requested withdraw amount's currency
secretKey	O	string		Pay card secret code (specific for certain withdraw methods)
transactionReference	M	string		Operator's transaction reference
note	O	string		User note
userIp	M	string		User IP (IPv4 format)
type	O	int		0 – Pending (by default), 1 – Skip pending
isMobile	O	int		"1" if it is mobile, otherwise "0"
iovationBlackBox	O	string		Device unique code
paymentMethod	O	string		Name of the payment method, used only for Trustly vendor. Value is "Trustly.PayNPlay" by default.

### Response variables

Name	M/O	Type	Size	Comments
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
transactionId		string		GMCore transaction identifier
balanceAmount		decimal		User's wallet balance amount after transaction (if available, returned in <i>RequestCurrency</i> ). A maximum of 2 decimals places can be used after the integer for proper functionality.
transStatus		int		GMCore transaction status.  Success = 1,  Failed = 2,  Pending = 4,  DebitFailed = 6,  CreditFailed = 8,  Cancelled = 9,  RollBack = 10,  PendingNotification = 11,  PendingApproval = 12
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

### Request example

<http://localhost/ServerAPI/Withdraw/1.0/testID/testCode>

```
{
  "sessionId": "64b27b1b-32ac-4cf4-ad06-293fbe435237",
  "userId": "23455322",
  "debitAccountId": 1524784,
  "creditPayCardId": 0,
  "creditPayCardVendorId": 1063,
  "requestAmount": 5,
  "requestCurrency": "EUR",
  "transactionReference": "d7ed4b5fde474e8e875ca588abb24bfe",
```

```

"userIp": "127.0.0.1",
"note": "Test withdraw note"
}

```

#### Response example

```

{
  "balanceAmount": "34.24",
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2016-06-27 13:51:06",
  "transStatus": 4,
  "transactionId": "1680119",
  "version": "1.0"
}

```

#### e) ProceedWithdraw

Changes a pending withdraw transaction into a successful one, by processing it and actually moving the money in the player bank account or payment vendor of choice.

URL: |GmServiceURL|/ProceedWithdraw/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	Data type			Comments
	M/O	Type	Size	
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
transactionReference	M	string		Operator's transaction reference

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
transactionId		string		GMCore transaction identifier

transStatus		int		GMCore transaction status.  Success = 1,  Failed = 2,  Pending = 4,  DebitFailed = 6,  CreditFailed = 8,  Cancelled = 9,  RollBack = 10,  PendingNotification = 11,  PendingApproval = 12
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

http://localhost/ServerAPI/ProceedWithdraw/1.0/testID/testCode

```
{
  "transactionReference": "d7ed4b5fde474e8e875ca588abb24bfe",
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2016-06-27 13:51:06",
  "transStatus": 1,
  "transactionId": "1680119",
  "version": "1.0"
}
```

#### f) RollbackWithdraw

Changes a pending withdraw transaction into a rolled back one, by returning it to the player's CasinoWallet.

URL: |GmServiceURL|/RollbackWithdraw/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	Data type			Comments
	M/O	Type	Size	

version	M	string		API version, set to "1.0"
partnerId	M	string		API partner identifier
partnerKey	M	string		API partner key
transactionReference	M/O	string		Operator's transaction reference.
preTransId	O/M	string		The pre-transaction ID of the withdraw.

**Note** - Request must use either transactionReference OR preTransId parameter, if both are present an error will be returned.

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
• errorCode		int		Error code
• errorDetails		string		Possible error details
• errorMessage		string		Error message
• logId		long		Error log ID in the system
transactionId		string		GMCore transaction identifier
transStatus		int		GMCore transaction status.  Success = 1,  Failed = 2,  Pending = 4,  DebitFailed = 6,  CreditFailed = 8,  Cancelled = 9,  RollBack = 10,  PendingNotification = 11,  PendingApproval = 12
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

http://localhost/ServerAPI/RollbackWithdraw/1.0/testID/testCode

```
{
  "transactionReference": "d7ed4b5fde474e8e875ca588abb24bfe",
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
  }
}
```

```

        "logId": 0
    },
    "success": 1,
    "timestamp": "2016-06-27 13:51:06",
    "transStatus": 1,
    "transactionId": "1680119",
    "version": "1.0"
}

```

#### Response example (preTransId and transactionReference both present)

```

{
    "errorData": {
        "errorCode": 1034,
        "errorDetails": [],
        "errorMessage": "There are too many parameters in the request. Please use only one of the following valid parameters: transactionReference, preTransID",
        "logId": 99204778
    },
    "success": 0,
    "timestamp": "2018-12-13 16:05:30",
    "transStatus": 0,
    "transactionId": null,
    "version": "1.0"
}

```

#### g) ProcessAsyncTrans

This method is used to finalize a transaction.

URL: |GmServiceURL|/ServerAPI/ProcessAsyncTrans/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
hc	M	string		GM_hc received from PaymentTrust
responseFields	M	array		Parameters received from PaymentTrust on postback URL. See method Deposit > postBackURL
*key	M	string		MD, PaRes, Password, cancelFlag
*value	M	string		MD, PaRes, Password, cancelFlag values
secretKey	M	string		Card security code
sid	M	string		GM_sid received from PaymentTrust
userIp	M	string	7 - 15	User IP (IPv4 format)

#### Response variables

Name	Data type	Size	Comments
balanceAmount	string		Gaming account balance after the transaction is completed. A maximum of 2 decimal places can be used after the integer for proper functionality.

errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
transactionId	string		Unique transaction identity number in the GamMatrix system
version	string		Version used, set to "1.0"

#### Request example

```
{
  "hc": "String content",
  "responseFields": [{
    "key": "String content",
    "value": "String content"
  }],
  "secretKey": "String content",
  "sid": "String content",
  "userIp": "String content"
}
```

#### Response example

```
{
  "balanceAmount": "String content",
  "errorData": {
    "errorCode": 2147483647,
    "errorDetails": ["String content"],
    "errorMessage": "String content",
    "logId": 9223372036854775807
  },
  "success": 2147483647,
  "timestamp": "String content",
  "transactionId": "String content",
  "version": "String content"
}
```

### h) GetUserTransactionHistory

Gets a list of user transactions for the specified period.

URL: |GmServiceURL|/ServerAPI/GetUserTransactionHistory/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
completedFrom	M	string		From date. Format: yyyy-mm-dd



completedTo	M	string		To date. Format: yyyy-mm-dd
sessionId	M	string	40	User's session ID. See method LoginUser for how to get SessionId

## Response variables

Name	Data type	Size	Comments
depositTransList	array		List of deposit transactions for the user in the selected period
*created	string		Date when the transaction was created. Format yyyy-MM-dd HH:mm:ss
*creditAccountId	long		Credited gaming account ID
*creditAmount	decimal		Credit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*creditCurrency	string		Credit currency
*creditDisplayName	string		Credit account display name
*debitDisplayName	string		Debited account display name
*debitPayItemName	string		Debited PayCard display name
*debitPayItemVendorName	string		Payment vendor name
*id	long		Transaction ID
*status	int		Setup = 0, Success = 1, Failed = 2, Processing = 3, Pending = 4
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"
withdrawTransList	array		List of withdraw transactions for the user in the selected period
*created	string		Date when the transaction was created. Format yyyy-MM-dd HH:mm:ss
*creditDisplayName	string		Credited account display name
*creditPayItemName	string		Credited PayCard name
*creditPayItemVendorName	string		Payment vendor name
*debitAccountId	long		Debited gaming account ID
*debitAmount	decimal		Debit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*debitCurrency	string		Debit currency
*debitDisplayName	string		Debited account display name
*id	long		Transaction ID
*status	int		Setup = 0, Success = 1, Failed = 2, Processing = 3, Pending = 4

## Request example

```
{
  "completedFrom":"2017-10-15",
  "completedTo":"2017-10-16",
  "sessionId":"N5iYNzwWQkShj/5MAmxVIT1tAoShWBRLcv685mO1dyc="
}
```

## Response example (No transactions)

```

{
  "depositTransList": null,
  "errorData": {
    "errorCode": 1000,
    "errorDetails": [],
    "errorMessage": "Object reference not set to an instance of an object.",
    "logId": 100019626
  },
  "success": 0,
  "timestamp": "2018-12-19 14:42:31",
  "version": "1.0",
  "withdrawTransList": null
}

```

#### Response example

```

{
  "depositTransList": [{
    "created": "String content",
    "creditAccountId": 9223372036854775807,
    "creditAmount": 12678967.54,
    "creditCurrency": "String content",
    "creditDisplayName": "String content",
    "debitDisplayName": "String content",
    "debitPayItemName": "String content",
    "debitPayItemVendorName": "String content",
    "id": 9223372036854775807,
    "status": 2147483647
  }],
  "errorData": {
    "errorCode": 2147483647,
    "errorDetails": ["String content"],
    "errorMessage": "String content",
    "logId": 9223372036854775807
  },
  "success": 2147483647,
  "timestamp": "String content",
  "version": "String content",
  "withdrawTransList": [{
    "created": "String content",
    "creditDisplayName": "String content",
    "creditPayItemName": "String content",
    "creditPayItemVendorName": "String content",
    "debitAccountId": 9223372036854775807,
    "debitAmount": 12678967.54,
    "debitCurrency": "String content",

```

```

        "debitDisplayName": "String content",

        "id": 9223372036854775807,

        "status": 2147483647

    }
}

```

## i) GetUserPendingWithdrawals

Returns a list of pending withdrawals for the user.

URL: |GmServiceURL|/ServerAPI/GetUserPendingWithdrawals/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
pendingWithdrawalsTransList	array		List of pending withdrawals for the user
*approvalStatus	Int	1	Possible values are: <ul style="list-style-type: none"> <li>• 0 – None</li> <li>• 1 – Pending for approval</li> <li>• 2 – Pending for pick up</li> </ul>
*created	DateTime		Date when the transaction was created. Format yyyy-MM-dd HH:mm:ss
*creditDisplayName	string		Credited account display name
*creditPayItemName	string		Credited PayCard name
*creditPayItemVendorName	string		Payment vendor name
*debitAccountId	long		Debited gaming account ID
*debitAmount	decimal		Debit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*debitCurrency	string		Debit currency
*debitDisplayName	string		Debited account display name
*id	long		Transaction ID
*status	int		Setup = 0, Success = 1, Failed = 2, Processing = 3, Pending = 4
*transactionReference	string		The generated transaction reference, necessary for processing withdrawals or rollbacks.
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Response example

```

{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },

```

```

"pendingWithdrawalsTransList": [{
    "approvalStatus": 2
    "created": "/Date(1556800350263+0000)/",
    "creditDisplayName": "Interac2Pay.BankTransfer",
    "creditPayItemName": " Interac2Pay.BankTransfer (MoneyMatrix)",
    "creditPayItemVendorName": "MoneyMatrix",
    "debitAccountId": 2167929,
    "debitAmount": 100,
    "debitCurrency": "EUR",
    "debitDisplayName": "CasinoWallet",
    "id": 6854775807,
    "status": 4
    "transactionReference": ""
}],
"success": 1,
"timestamp": "2019-05-16 14:36:57",
"version": "1.0"
}

```

## j) UpdatePayCardStatus

Changes the status of user's Paycard.

URL: |GmServiceURL|/ServerAPI/UpdatePayCardStatus/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

### Request variables

Name	M/O	Data type	Size	Comments
newStatus	M	int		InActive = 1, Active = 0. Blocked = 1, Closed = 2
payCardId	M	long		Users's paycard ID to be updated. See method GetUserPayCards
sessionId	M	string	40	User's session ID. See method LoginUser for how to get SessionId

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Request example

```

{
    "newStatus": 1,

```

```

"payCardId":245323,

"sessionId":"a1252478-5873-47e0-9d6e-fe00ce2e8ca2"

}

```

#### Response example

```

{

  "errorData": {

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

  },

  "success": 1,

  "timestamp": "2018-08-07 13:52:54",

  "version": "1.0"

}

```

### m) Transfer

Transfers from CasinoWallet to ExternalWallet 3rd Party.  
 URL: |GmServiceURL|/ServerAPI/Transfer/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
version	M	string		API version, set to “1.0”
PARTNERID	M	string		API partner identifier
PARTNERKEY	M	string		API partner key

#### Request variables body

Name	M/O	Data type	Size	Comments
sessionId	M	string		User's session ID. See method LoginUser for how to get SessionId
direction	M	int	1	0 - debit from CasinoWallet, 1 - credit to CasinoWallet
amount	M	int		Amount to be deposited, maximum value: 1 million, with "." decimal symbol for separation. A maximum of 2 decimals must be used after the integer for proper functionality.
currency	M	string	3	Currency of the transfer, ISO-4217 three char code
bonusVendorId	O	int		Bonus Vendor ID, should be set as default 112 (CasinoWallet)
bonusCode	O	string		Bonus Code, only applicable for <i>direction</i> equal 1 (Credit)

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), ErrorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	int		Error log ID in the system

transactionId	string		The ID of the transaction
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"
warning	string		Warning

#### Request example

```
{
  "amount":1500,
  "bonusCode":"test #Deleted 26/09/2012#1000090",
  "bonusVendorId":112,
  "currency":"TRY",
  "direction":1,
  "sessionId":"ae846f35-8ed0-40d3-a50f-6f31fff2b9c2"
}
```

#### Response example (successful response)

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-07 13:45:19",
  "version": "1.0"
}
```

#### Response example (failed response - session not found)

```
{
  "errorData":{
    "errorCode": 1004,
    "errorDetails": [],
    "errorMessage": "Session not found",
    "logId": 736694
  },
  "success":0,
  "timestamp":"2018-08-07 13:47:07",
  "transactionId": null,
  "version":"1.0",
  "warning":"null"
}
```

#### Response example (failed response - bonus code not authorized)

```
{
  "errorData":{
    "errorCode": 1004,
    "errorDetails": [],
    "errorMessage": "The bonus code you have entered is not recognized. Please check the code and try again (Casino bonus not found, code: test #Deleted 26/09/2012#1000091)",
    "logId": 736694
  },
  "success":0,
  "timestamp":"2018-08-07 13:47:07",
  "transactionId": null,
  "version":"1.0",
  "warning":"null"
}
```

#### Response example (failed response - bonus vendor not valid)

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2016-07-15 10:43:31",
  "transactionId": "712578",
  "version": "1.0",
  "warning": "Casino bonus 1000090 cannot be applied for user 112912 depositing EUR 100.21: User does not match user level"
}
```

#### Response example (failed response - currency rate is missing)

```
{
  "errorData": {
    "errorCode": 1025,
    "errorDetails": [],
    "errorMessage": "bonusVendorID 'PlaynGO' is not valid, the only allowed bonus vendor in this method is CasinoWallet (112)",
    "logId": 736699
  },
  "success": 0,
```

```

    "timestamp": "2016-07-15 10:50:13",

    "transactionId": null,

    "version": "1.0",

    "warning": null

}

```

## n) GetUserTransactionList

Gets a list of user transactions for the specified period.

URL: |GmServiceURL|/ServerAPI/GetUserTransactionList/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

### Request variables

Name	Data type	Size	Comments
completedFrom	string		Search transactions starting from selected date. Format: <b>yyyy-mm-dd hh:mm:ss</b>
completedTo	string		Search transactions up until selected date. Format: <b>yyyy-mm-dd hh:mm:ss</b>
sessionId	string	40	Player's session identifier. See <code>LoginUser</code> method on how to get <code>SessionId</code> .

### Response variables

Name	Data type	Size	Comments
depositTransList	List of Object		List of deposit transactions for the user in the selected period
*created	string		Date when the transaction was created. Format yyyy-MM-dd HH:mm:ss
*creditAccountId	long		Credited gaming account ID
*creditAmount	decimal		Credit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*creditCurrency	string		Credit currency
*creditDisplayName	string		Credit account display name
*debitDisplayName	string		Debited account display name
*debitPayItemName	string		Debited PayCard display name
*debitPayItemVendorName	string		Payment vendor name
*id	long		Transaction ID
*status	int		Setup = 0, Success = 1, Failed = 2, Processing = 3, Pending = 4
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	list of string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int		1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"
withdrawTransList	List of object		List of withdraw transactions for the user in the selected period
*created	string		Date when the transaction was created. Format yyyy-MM-dd HH:mm:ss
*creditDisplayName	string		Credited account display name
*creditPayItemName	string		Credited PayCard name
*creditPayItemVendorName	string		Payment vendor name
*debitAccountId	long		Debited gaming account ID
*debitAmount	decimal		Debit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*debitCurrency	string		Debit currency



*debitDisplayName	string		Debited account display name
*id	long		Transaction ID
*status	int		Setup = 0, Success = 1, Failed = 2, Processing = 3, Pending = 4

#### Request example (only for POST methods)

```
{
  "completedFrom": "2015-09-15",
  "completedTo": "2018-10-16",
  "sessionId": "bca9431d-8017-4554-a385-f6dcfd51197f"
}
```

#### Response example

```
{
  "depositTransList": [{
    "created": "2015-10-14 06:56:34",
    "creditAccountId": 2155248,
    "creditAmount": 20.00,
    "creditCurrency": "EUR",
    "creditDisplayName": "CasinoWallet",
    "debitDisplayName": "Credit Card",
    "debitPayItemName": "Credit Card",
    "debitPayItemVendorName": "PaymentTrust",
    "id": 1543134661,
    "status": 1
  }],
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2015-10-15 08:39:20",
  "version": "1.0",
  "withdrawTransList": [{
    "created": "2015-10-14 06:56:34",
    "debitAccountId": 2155248,
    "debitAmount": 20.00,
    "debitCurrency": "EUR",
    "creditDisplayName": "CasinoWallet",
    "debitDisplayName": "Credit Card",
    "creditPayItemName": "Credit Card",
    "creditPayItemVendorName": "PaymentTrust",
  ]
}
```

```

    "id": 1543134661,

    "status": 1

  },

}

```

## o)DepositAndPlay

Performs DepositAndPlay call to Trustly. PayNPlay payment vendor,

URL: URL. |GmServiceURL|/DepositAndPlay/1.0/{Partner ID}/{Partner KEY

HTTP Method: POST

### Request variables

Name	M/O	Type	Size	Comments
paymentMethod	M	string		Should be "Trustly.PayNPlay"
ipAddress	M	string		IP address of user
countryCode	O	string	2	Country code of user
successUrl	M	string		URL to redirect user in case of success
failUrl	M	string		URL to redirect user in case of error
cancelUrl	M	string		URL to redirect user in case of cancel
amount	O	decimal		Amount of deposit. A maximum of 2 decimal places can be used after the integer for proper functionality.
btag	O	string		Affiliate referral id
currency	O	string	3	Currency of deposit
language	O	string		Player's preferred language to display Trustly user interface. Value will have ISO 3166-1-alpha-2 format.
isMobile	O	int		Informs if the player is using a mobile device. Sends the following values  1 for mobile device  0 for other
sessionId	O	string		The player's session ID, generated when logging in to his account.

### Response variables

Name	M/O	Type	Size	Comments
errorData	M	Object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	M	int		Code of error
*errorDetails	M	list<string>		Details of error
*errorMessage	M	string		Error message
*logId	M	int		GmCore error log ID
redirectUrl	M	string		URL to redirect user
success	M	int		Success flag: 1 – true, 0 - false
timestamp	M	string		Method call timestamp in format yyyy-MM-dd HH:mm:ss
version	M	string		Version used, set to "1.0"

### Request example

```

{

  "amount":15,

  "btag":"AK",

  "cancelUrl":"https://cancelUrl.com",

}

```

```

"countryCode":"RO",
"currency":"EUR",
"failUrl":"https://cancelUrl.com",
"ipAddress":"127.0.0.1",
"paymentMethod":" Trustly.PayNPlay ",
"successUrl":"https://cancelUrl.com",
"language":"EN",
"isMobile": 1
"sessionId": "5742859e1959ef91c0e89e4sa34f8672f1e93293dbeef0054",
}

```

#### Response example

```

{
  "errorData":{
    "errorCode":0,
    "errorDetails":[],
    "errorMessage":"",
    "logId":0
  },
  "redirectUrl":" https://redirect.moneymatrix.com/Process/Redirect/Trustly.PayNPlay/Transaction/Nonce ",
  "status":1,
  "success":1,
  "timestamp":"2017-09-25 14:12:56",
  "version":"1.0"
}

```

## p)LoginAndPlay

Performs LoginAndPlay call to Trustly. PayNPlay payment vendor,

URL: URL. |GmServiceURL|/LoginAndPlay/1.0/{Partner ID}/{Partner KEY

HTTP Method: POST

#### Request variables

Name	M/O	Type	Size	Comments
paymentMethod	M	string		Should be "Trustly.PayNPlay"
ipAddress	M	string		IP address of user
countryCode	O	string	2	Country code of user
successUrl	M	string		URL to redirect user in case of success
failUrl	M	string		URL to redirect user in case of error
cancelUrl	M	string		URL to redirect user in case of cancel
btag	O	string		Affiliate referral id
language	O	string		Player's preferred language to display Trustly user interface. Value will have ISO 3166-1-alpha-2 format.

isMobile	O	int		<p>Informes if the player is using a mobile device. Sends the following values</p> <p>1 for mobile device</p> <p>0 for other</p>
sessionId	O	string		The player's session ID, generated when logging in to his account.

#### Response variables

Name	M/O	Type	Size	Comments
errorData	M	Object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	M	int		Code of error
*errorDetails	M	list<string>		Details of error
*errorMessage	M	string		Error message
*logId	M	int		GmCore error log ID
redirectUrl	M	string		URL to redirect user
success	M	int		Success flag: 1 – true, 0 - false
timestamp	M	string		Method call timestamp in format yyyy-MM-dd HH:mm:ss
version	M	string		Version used, set to "1.0"

#### Request example

```
{
  "paymentMethod": "Trustly.PayNPlay",
  "ipAddress": "127.0.0.1",
  "countryCode": "SE",
  "successUrl": "https://www.w3schools.com/action_page.php",
  "failUrl": "https://www.w3schools.com/action_page.php",
  "cancelUrl": "https://www.w3schools.com/action_page.php",
  "btag": "AK",
  "language": "EN",
  "isMobile": 0
  "sessionId": "5742859e1959ef91c0e89e4284f8672f1e93293dbeef0054"
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "redirectUrl": "https://redirect.moneymatrix.com/Process/Redirect/Trustly.PayNPlay/SelectCustomer/Nonce",
  "success": 1,
  "timestamp": "2018-05-16 13:03:29",
}
```

"version": "1.0"

}

## q)GetBalance

This method returns the complete balance of a player, meaning real money, locked money and bonus money for each type of wallet, to be displayed on the front end.

URL: |GmServiceURL|/ServerAPI/GetBalance/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID} /{PRODUCTCATEGORY}

HTTP Method: GET

### Request variables

Name	M /O	Type	Size	Comments
sessionid	M	string	40	User session ID. See method LoginUser for how to get SessionId
productCategory	O	enum		Specifies for what product the request will retrieve and display bonus money.If parameter is missing, method will retrieve all bonus money wallets(same result as Unspecified). Possible values are: <ul style="list-style-type: none"><li>• 0 - Unspecified.</li><li>• 1 - Casino</li><li>• 2 - Sports</li></ul>

### Response variables

Name	M /O	Type	Size	Comments
errorData	M	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	M	int		Error code
*errorDetails	M	string		Possible error details
*errorMessage	M	string		Error message
*logId	M	long		Error log ID in the system
success	M	int		If successfully called returns "1", else "0"
Timestamp	M	datetime		Date and time of the request
Version	M	string		Version used, set to "1.0"
wallets	M	array		
*bonusMoney	M	decimal		Amount of bonus money available on the player's account. A maximum of 2 decimal places can be used after the integer for proper functionality.
*bonusMoneyCurrency	M	string		Currency used for the bonus funds wallet
*lockedMoney	M	decimal		Amount of locked money on the player's account due to bonuses. A maximum of 2 decimal places can be used after the integer for proper functionality.
*lockedMoneyCurrency	M	string		Currency used for the locked money amounts
*name	M	string		For MainWallet name will be "MainWallet" and for Non-seamless wallets will be "VendorID+Wallet"
*realMoney	M	decimal		Amount of real money available on the player's account. A maximum of 2 decimal places can be used after the integer for proper functionality.
*realMoneyCurrency	M	string		Currency used for the real money wallet

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": ""
```

```

    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-02 10:52:12",
  "version": "1.0",
  "wallets": [
    {
      "bonusMoney": 0,
      "bonusMoneyCurrency": "EUR",
      "lockedMoney": 0,
      "lockedMoneyCurrency": "EUR",
      "name": "MainWallet",
      "realMoney": 0,
      "realMoneyCurrency": "EUR"
    },
    {
      "bonusMoney": 10,
      "bonusMoneyCurrency": "EUR",
      "lockedMoney": 0,
      "lockedMoneyCurrency": "EUR",
      "name": "MainWallet",
      "realMoney": 9893.42,
      "realMoneyCurrency": "EUR"
    }
  ]
}

```

### r)GetDepositMaxAmount

This method returns the maximum amount of money a player can deposit based on the limits he has established  
 URL: *|GmServiceURL|/GetDepositMaxAmount/{version}/{partnerID}/{partnerKey}*

HTTP Method: POST

#### Request variables

Name	M/O	Type	Size	Comments
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	O	string		User session identifier
userId	O	long		User identifier
currency	O	string		Currency to be used in the response

## Response variables

Name	M/O	Type	Size	Comments
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"
errorData		object		If success:  errorCode = 0  logId = 0  errorDetails = [] (empty)  errorMessage = "" (empty)
· errorCode		int		Error code
· errorDetails		string		Possible error details
· errorMessage		string		Error message
· logId		long		Error log ID in the system
currency		string	3	Currency, matches either request currency if passed or user's main wallet currency
amount		decimal		Deposit max amount. A maximum of 2 decimal palces can be used after the integer for proper functionality.

## Request example

http://localhost/ServerAPI/GetDepositMaxAmount/1.0/JetId/JetCode

```
{  
  "currency": "EUR",  
  "userId": "2005456"  
}
```

## Response example

```
{  
  "amount": 500,  
  "currency": "EUR",  
  "errorData": {  
    "errorCode": 0,  
    "errorDetails": [],  
    "errorMessage": "",  
    "logId": 0  
  },  
  "success": 1,  
  "timestamp": "2018-07-17 12:41:51",  
  "version": "1.0"  
}
```

## Response if user is blocked:

```
{
```

```

"amount": 0,
"currency": null,
"errorData": {
  "errorCode": 1006,
  "errorDetails": [],
  "errorMessage": "User account blocked",
  "logId": 180738
},
"success": 0,
"timestamp": "2018-07-17 12:42:22",
"version": "1.0"
}

```

#### Response if limit is reached:

```

{
  "amount": 0,
  "currency": null,
  "errorData": {
    "errorCode": 1032,
    "errorDetails": [],
    "errorMessage": "Deposit limit is zero",
    "logId": 180740
  },
  "success": 0,
  "timestamp": "2018-07-17 12:42:48",
  "version": "1.0"
}

```

#### s)GetUserTransactionHistory\_RecentGaming

Retrieves all transaction history for the last 24 hours and 59 minutes at most directly from GMCore. If a player inputs no time frame filters, this method will be called directly.

URL: /ServerAPI/GetUserTransactionHistory\_RecentGaming/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: **POST**

Name	Data type	Size	Comments
completedFrom	String		From date. Format: yyyy-mm-dd hh:mm:ss
completedTo	String		To Date. Format: yyyy-mm-dd hh:mm:ss
pageIndex	Int		Starting page in transaction history. Value is always 0 to show all results from latest transactions to oldest.
pageSize	Int		Number of transactions displayed per page.
sessionId	String		The player's unique session ID.
userId	String		The player's unique ID in GamMatrix.



**Request Example:**

```
{  
  "completedFrom": "2019-04-23 00:05:59",  
  "completedTo": "2019-04-24 00:00:59",  
  "pageIndex": 0,  
  "pageSize": 100,  
  "sessionId": "23dh46346-45gsdrfg-ejtg8556",  
  "userId": 33399124  
}
```

**Response Parameters:**

Name	Data type	Size	Comments
transList	Object		Object containing transaction details.
*created	Datetime		Transaction starting time. Format: yyyy-mm-dd hh:mm:ss
*creditAccountId	Long		User's credit account identifier. Received from <i>GetUserAccounts</i> API.
*creditAmount	Decimal		Credit amount.
*creditCurrency	String		Credit currency
*creditDisplayName	String		Credited account display name
*creditPayItemName	String		Credited PayCard name
*debitAccountId	Long		Debited gaming account ID
*debitAmount	Decimal		Debit amount
*debitCurrency	String		Debit currency
*debitDisplayName	String		Debited account display name
*debitPayItemName	String		Debited PayCard display name
*debitPayItemVendorName	String		Payment vendor name.
*finished	Datetime		Transaction completion date. Format: yyyy-mm-dd hh:mm:ss
*id	Long		Pretrans ID of the transactions.
*status	Int		GMCore transaction status.  · Success = 1 · Failed = 2 · Pending = 4 · DebitFailed = 6 · CreditFailed = 8 · Cancelled = 9 · RollBack = 10 · PendingNotification = 11 · PendingApproval = 12
*transId	Long		Unique ID of the transaction.

*transType	String		Transaction type. Can be any of the following: <ul style="list-style-type: none"> <li>· Deposit</li> <li>· Withdraw</li> <li>· Transfer</li> <li>· User2user</li> <li>· User2vendor</li> <li>· Vendor2user</li> <li>· WalletDebit</li> <li>· WalletCredit</li> </ul>
errorData	Object		If success, <ul style="list-style-type: none"> <li>· errorCode = 0</li> <li>· logId = 0</li> <li>· errorDetails = "" (empty)</li> <li>· errorMessage = "" (empty)</li> </ul>
*errorCode	Int		Error code
*errorDetails	String		Possible error details
*errorMessage	String		Error message
*logId	Long		Error log ID in the system
success	Int	1	<ul style="list-style-type: none"> <li>· 1 - success</li> <li>· 0 - failed</li> </ul>
timestamp	String		yyyy-MM-dd HH:mm:ss
version	String		Version used, set to "1.0"

### Response Example

```

"errorData": {
  "errorCode": 0,
  "errorDetails": [],
  "errorMessage": "",
  "logId": 0
},
"success": 1,
"timestamp": "2019-04-23 08:26:01",
"totalCount": 0,
"version": "1.0",
"transList": [
  {
    "created": "2019-04-14 08:32:48",
    "creditAccountId": 10398970,
    "creditAmount": 33,
    "creditCurrency": "EUR",
    "creditDisplayName": "CasinoWallet",
  }
]

```

```

    "creditPayItemName": "CasinoWallet",
    "creditPayItemVendorName": "CasinoWallet",
    "debitAccountId": 0,
    "debitAmount": 33,
    "debitCurrency": "EUR",
    "debitDisplayName": "PPro.TrustPay",
    "debitPayItemName": "PPro.TrustPay (MoneyMatrix) ",
    "debitPayItemVendorName": "MoneyMatrix",
    "finished": "2019-04-14 08:33:32",
    "id": 1566932425,
    "status": 1,
    "transId": 1561685446,
    "transType": "Deposit",
}

```

#### t)GetUserTransactionHistory\_Gaming

Retrieves a player's transaction history regarding wallet debit and wallet credit money movements. Can retrieve data up to 12 months old from databases.

URL: /ServerAPI/GetUserTransactionHistory\_Gaming/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: **POST**

Name	Data type	Size	Comments
completedFrom	String		From date. Format: yyyy-mm-dd hh:mm:ss
completedTo	String		To Date. Format: yyyy-mm-dd hh:mm:ss
pageIndex	Int		Starting page in transaction history. Value is always 0 to show all results from latest transactions to oldest.
pageSize	Int		Number of transactions displayed per page.
sessionId	String		The player's unique session ID.
userId	String		The player's unique ID in GamMatrix.

#### Request Example:

```

{
  "completedFrom": "2019-01-10 00:05:59",
  "completedTo": "2019-04-27 00:00:00",
  "pageIndex": 0,
  "sessionId": "54g447-4707-877ckli8-ds3425d",
  "userId": 3339914
}

```

#### Response Parameters:

Name	Data Type	Size	Comments
------	-----------	------	----------

transList	Object		Object containing transaction details.
*created	Datetime		Transaction starting time. Format: yyyy-mm-dd hh:mm:ss
*creditAccountId	Long		User's credit account identifier. Received from <i>GetUserAccounts</i> API.
*creditAmount	Decimal		Credit amount.
*creditCurrency	String		Credit currency
*creditDisplayName	String		Credited account display name
*creditPayItemName	String		Credited PayCard name
*debitAccountId	Long		Debited gaming account ID
*debitAmount	Decimal		Debit amount
*debitCurrency	String		Debit currency
*debitDisplayName	String		Debited account display name
*debitPayItemName	String		Debited PayCard display name
*debitPayItemVendorName	String		Payment vendor name.
*finished	Datetime		Transaction completion date. Format: yyyy-mm-dd hh:mm:ss
*id	Long		Pretrans ID of the transactions.
*status	Int		<p>GMCore transaction status.</p> <ul style="list-style-type: none"> <li>· Success = 1</li> <li>· Failed = 2</li> <li>· Pending = 4</li> <li>· DebitFailed = 6</li> <li>· CreditFailed = 8</li> <li>· Cancelled = 9</li> <li>· RollBack = 10</li> <li>· PendingNotification = 11</li> <li>· PendingApproval = 12</li> </ul>
*transId	Long		Unique ID of the transaction.
*transType	String		<p>Transaction type. Can be any of the following:</p> <ul style="list-style-type: none"> <li>· WalletCredit</li> <li>· WalletDebit</li> </ul>
errorData	Object		<p>If success,</p> <ul style="list-style-type: none"> <li>· errorCode = 0</li> <li>· logId = 0</li> <li>· errorDetails = "" (empty)</li> <li>· errorMessage = "" (empty)</li> </ul>
*errorCode	Int		Error code
*errorDetails	String		Possible error details
*errorMessage	String		Error message
*logId	Long		Error log ID in the system
success	Int	1	<ul style="list-style-type: none"> <li>· 1 - success</li> <li>· 0 - failed</li> </ul>
timestamp	String		yyyy-MM-dd HH:mm:ss
version	String		Version used, set to "1.0"

#### Response Example

```

"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
"success": 1,
"timestamp": "2019-04-23 08:26:01",
"totalCount": 0,
"version": "1.0",
"transList": [
    {
        "created": "2019-04-17 08:30:48",
        "creditAccountId": 10398970,
        "creditAmount": 10,
        "creditCurrency": "EUR",
        "creditDisplayName": "@NetEnt",
        "creditPayItemName": "System",
        "creditPayItemVendorName": "System",
        "debitAccountId": 0,
        "debitAmount": 10,
        "debitCurrency": "EUR",
        "debitDisplayName": "CasinoWallet",
        "debitPayItemName": "CasinoWallet",
        "debitPayItemVendorName": "CasinoWallet",
        "finished": "2019-04-17 08:30:49",
        "id": 1566932489,
        "status": 1,
        "transId": 6191685446,
        "transType": "WalletDebit",
    }
],

```

#### u)GetUserTransactionHistory\_Payment

Retrieves the player's transaction history regarding deposits, withdrawals, transfers and user2user money movements. Can retrieve data up to 12 months old from databases.

URL: /ServerAPI/GetUserTransactionHistory\_Payment/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: **POST**

Name	Data type	Size	Comments
completedFrom	String		From date. Format: yyyy-mm-dd hh:mm:ss
completedTo	String		To Date. Format: yyyy-mm-dd hh:mm:ss
pageIndex	Int		Starting page in transaction history. Value is always 0 to show all results from latest transactions to oldest.
pageSize	Int		Number of transactions displayed per page.
sessionId	String		The player's unique session ID.
userId	String		The player's unique ID in GamMatrix.

#### Request Example:

```
{
  "completedFrom": "2019-04-10 00:59:59",
  "completedTo": "2019-04-27 00:00:00",
  "pageIndex": 0,
  "sessionId": "54g447-4707-877ckli8-ds3425d",
  "userId": 3339914
}
```

#### Response Parameters:

Name	Data Type	Size	Comments
transList	Object		Object containing transaction details.
*created	Datetime		Transaction starting time. Format: yyyy-mm-dd hh:mm:ss
*creditAccountId	Long		User's credit account identifier. Received from <i>GetUserAccounts</i> API.
*creditAmount	Decimal		Credit amount.
*creditCurrency	String		Credit currency
*creditDisplayName	String		Credited account display name
*creditPayItemName	String		Credited PayCard name
*debitAccountId	Long		Debited gaming account ID
*debitAmount	Decimal		Debit amount
*debitCurrency	String		Debit currency
*debitDisplayName	String		Debited account display name
*debitPayItemName	String		Debited PayCard display name
*debitPayItemVendorName	String		Payment vendor name.
*finished	Datetime		Transaction completion date. Format: yyyy-mm-dd hh:mm:ss
*id	Long		Pretrans ID of the transactions.

*status	Int		<p>GMCore transaction status.</p> <ul style="list-style-type: none"> <li>· Success = 1</li> <li>· Failed = 2</li> <li>· Pending = 4</li> <li>· DebitFailed = 6</li> <li>· CreditFailed = 8</li> <li>· Cancelled = 9</li> <li>· RollBack = 10</li> <li>· PendingNotification = 11</li> <li>· PendingApproval = 12</li> </ul>
*transId	Long		Unique ID of the transaction.
*transType	String		<p>Transaction type. Can be any of the following:</p> <ul style="list-style-type: none"> <li>· Deposit</li> <li>· Withdraw</li> <li>· Transfer</li> <li>· User2user</li> </ul>
errorData	Object		<p>If success,</p> <ul style="list-style-type: none"> <li>· errorCode = 0</li> <li>· logId = 0</li> <li>· errorDetails = "" (empty)</li> <li>· errorMessage = "" (empty)</li> </ul>
*errorCode	Int		Error code
*errorDetails	String		Possible error details
*errorMessage	String		Error message
*logId	Long		Error log ID in the system
success	Int	1	<ul style="list-style-type: none"> <li>· 1 - success</li> <li>· 0 - failed</li> </ul>
timestamp	String		yyyy-MM-dd HH:mm:ss
version	String		Version used, set to "1.0"

```

"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
"success": 1,
"timestamp": "2019-04-23 08:26:01",
"totalCount": 0,
"version": "1.0",
"transList": [
    {

```

```

    "created": "2019-04-14 08:32:48",

    "creditAccountId": 10398970,

    "creditAmount": 10,

    "creditCurrency": "EUR",

    "creditDisplayName": "Credit Card",

    "creditPayItemName": "WorldPay (MoneyMatrix)",

    "creditPayItemVendorName": "MoneyMatrix",

    "debitAccountId": 0,

    "debitAmount": 10,

    "debitCurrency": "EUR",

    "debitDisplayName": "CasinoWallet",

    "debitPayItemName": "CasinoWallet",

    "debitPayItemVendorName": "CasinoWallet",

    "finished": "2019-04-15 09:21:05",

    "id": 1566932489,

    "status": 1,

    "transId": 6191685446,

    "transType": "Withdraw",

  },

```

#### v)GetUserTransactionHistory\_Vendor

Retrieves the player's transaction history regarding user2vendor and vendor2user money movements. Can retrieve data up to 12 months old from databases

URL: /ServerAPI/GetUserTransactionHistory\_Vendor/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: **POST**

Name	Data type	Size	Comments
completedFrom	String		From date. Format: yyyy-mm-dd hh:mm:ss
completedTo	String		To Date. Format: yyyy-mm-dd hh:mm:ss
pageIndex	Int		Starting page in transaction history. Value is always 0 to show all results from latest transactions to oldest.
pageSize	Int		Number of transactions displayed per page.
sessionId	String		The player's unique session ID.
userId	String		The player's unique ID in GamMatrix.

#### Request Example:

```

{

  "completedFrom": "2019-04-10 00:59:59",

  "completedTo": "2019-04-27 00:00:00",

  "pageIndex": 0,

  "sessionId": "54g447-4707-877ckli8-ds3425d",

```



"userId": 3339914

}

#### Response Parameters:

Name	Data Type	Size	Comments
transList	Object		Object containing transaction details.
*created	Datetime		Transaction starting time. Format: yyyy-mm-dd hh:mm:ss
*creditAccountId	Long		User's credit account identifier. Received from <i>GetUserAccounts</i> API.
*creditAmount	Decimal		Credit amount.
*creditCurrency	String		Credit currency
*creditDisplayName	String		Credited account display name
*creditPayItemName	String		Credited PayCard name
*debitAccountId	Long		Debited gaming account ID
*debitAmount	Decimal		Debit amount
*debitCurrency	String		Debit currency
*debitDisplayName	String		Debited account display name
*debitPayItemName	String		Debited PayCard display name
*debitPayItemVendorName	String		Payment vendor name.
*finished	Datetime		Transaction completion date. Format: yyyy-mm-dd hh:mm:ss
*id	Long		Pretrans ID of the transactions.
*status	Int		GMCore transaction status. <ul style="list-style-type: none"><li>· Success = 1</li><li>· Failed = 2</li><li>· Pending = 4</li><li>· DebitFailed = 6</li><li>· CreditFailed = 8</li><li>· Cancelled = 9</li><li>· RollBack = 10</li><li>· PendingNotification = 11</li><li>· PendingApproval = 12</li></ul>
*transId	Long		Unique ID of the transaction.
*transType	String		Transaction type. Can be any of the following: <ul style="list-style-type: none"><li>· User2Vendor</li><li>· Vendor2User</li></ul>
errorData	Object		If success, <ul style="list-style-type: none"><li>· errorCode = 0</li><li>· logId = 0</li><li>· errorDetails = "" (empty)</li><li>· errorMessage = "" (empty)</li></ul>
*errorCode	Int		Error code
*errorDetails	String		Possible error details
*errorMessage	String		Error message
*logId	Long		Error log ID in the system

success	Int	1	<ul style="list-style-type: none"> <li>1 - success</li> <li>0 - failed</li> </ul>
timestamp	String		yyyy-MM-dd HH:mm:ss
version	String		Version used, set to "1.0"

```

"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
"success": 1,
"timestamp": "2019-04-23 08:26:01",
"totalCount": 0,
"version": "1.0",
"transList": [
    {
        "created": "2019-04-15 13:30:28",
        "creditAccountId": 10398970,
        "creditAmount": 10,
        "creditCurrency": "EUR",
        "creditDisplayName": "@UBS",
        "creditPayItemName": "System",
        "creditPayItemVendorName": "System",
        "debitAccountId": 0,
        "debitAmount": 10,
        "debitCurrency": "EUR",
        "debitDisplayName": "CasinoWallet",
        "debitPayItemName": "CasinoWallet",
        "debitPayItemVendorName": "CasinoWallet",
        "finished": "2019-04-15 13:30:29",
        "id": 1566932489,
        "status": 1,
        "transId": 2519685446,
        "transType": "User2Vendor",
    }
],

```

## 6. Responsible gambling

### a) SetDepositLimit

Sets deposit limit for a user.

URL: |GmServiceURL|/ServerAPI/SetDepositLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}/{AMOUNT}/{CURRENCY}

HTTP Method: GET

#### Request variables

Name	M / O	Type	Size	Comments
sessionid	M	string	40	User session ID. See method LoginUser for how to get SessionId.
period	M	int	1	Possible values are:  1 - daily limit  2 - weekly limit  3 - monthly limit.
amount	M	decimal		Maximum amount of money to be deposited. A maximum of 2 decimal places can be used after the integer for proper functionality.
currency	M	string		Currency used for the deposit limit.

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isScheduled	int		Flag indicating whether deposit limit is already set. 1 - true, 0 - false
success	int	1	1 - success, 0 - failed
timeStamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

|GmServiceURL|/ServerAPI/SetDepositLimit/1.0/{PARTNERID}/{PARTNERKEY}/83076b19-885b-4557-b05f-d929d456184f/1/100/EUR

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
  "success": 1,
  "timestamp": "2018-08-20 13:01:45",
  "version": "1.0"
}
```

#### b) GetUserRgDepositLimit

Returns the deposit limits set for the user.

URL: |GmServiceURL|/ServerAPI/GetUserRgDepositLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
depositLimitData	object		Deposit limit data
*amount	decimal		Deposit limit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.

*currency	string		Deposit limit currency
*period	int		0 - none, 1 - daily, 2 - weekly, 3 - monthly (if none is selected, an error will appear)
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example (no limits)

```
{
  "depositLimitData": null,
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-12-19 15:19:34",
  "version": "1.0"
}
```

#### Response example

```
{
  "depositLimitData": {
    "amount": 1000,
    "currency": "EUR",
    "period": 1
  },
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-12-19 15:22:26",
  "version": "1.0"
}
```

### c) ApproveDepositLimit

Approves the deposit limits set for the user.

URL: |GmServiceURL|/ServerAPI/ApproveDepositLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}\_

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 08:05:38",
  "version": "1.0"
}
```

### d) CancelDepositLimit

Cancels the deposit limits set for the user.

URL: |GmServiceURL|/ServerAPI/CancelDepositLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}

HTTP Method: GET

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 08:10:05",
  "version": "1.0"
}
```

### e) SetUserRgSelfExclusion

Sets self-exclusion period for a user.

URL: |GmServiceURL|/ServerAPI/SetUserRgSelfExclusion/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
period	M	int		SelfExclusionNone = 0, SelfExclusionFor7Days = 1 - will be converted to a cool off of 7 days (CoolOffFor7Days = 11), SelfExclusionFor30Days = 2 - will be converted to a cool off of 30 days (CoolOffFor30Days = 12), SelfExclusionFor3Months = 3 - will be converted to a cool off of 3 months (CoolOffFor3Months = 13), SelfExclusionFor6Months = 4, SelfExclusionFor1Year = 5, SelfExclusionPermanent = 6, SelfExclusionUntilSelectedDate = 7, SelfExclusionFor5Years = 8,
sessionId	M	string		User's session ID. See method LoginUser for how to get SessionId
SendEmailNotification	O	boolean		Whether to send an email notification to the player that he is being transferred to self-excluded state
ExpiryDate	O	string		The date at which the self exclusion period ends. This should only be set if the period parameter has the value 7 (SelfExclusionUntilSelectedDate), and should be a date larger than 6 months from the current date (Self Exclusion cannot be smaller than 6 months). The value needs to be in the <b>JSON</b> <b>Serializer format</b> (milliseconds since the beginning of the Unix Epoch with a TimeZone) and a Epoch Converter needs to be used.
userId	O	string		The user ID of the player about to be selfexcluded. Parameter can be used when the operator receives a request for self-exclusion from one of their players to skip logging in to the players' account.

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```
{
  "ExpiryDate": "VDate(928149600000+0000)V",
  "SendEmailNotification": true,
  "period": 1,
  "sessionId": "a1252478-5873-47e0-9d6e-fe00ce2e8ca2"
  "userId": "3317449"
}
```

#### Response example

```
{
  "errorData": {
```

```

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

  },

  "success": 1,

  "timestamp": "2018-08-07 13:32:57",

  "version": "1.0"

}

```

#### f) GetUserRgSelfExclusion

Checks if user set a self-exclusion or cool-off period.

URL:|GmServiceURL|/ServerAPI/GetUserRgSelfExclusion/{VERSION}/{PARTNERID}/{PARTNERKEY}/{USERID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
period	int		Possible values are: <ul style="list-style-type: none"> <li>SelfExclusionNone = 0,</li> <li>SelfExclusionFor7Days = 1,</li> <li>SelfExclusionFor30Days = 2,</li> <li>SelfExclusionFor3Months = 3,</li> <li>SelfExclusionFor6Months = 4,</li> <li>SelfExclusionFor1Year = 5,</li> <li>SelfExclusionPermanent = 6,</li> <li>SelfExclusionUntilSelectedDate = 7,</li> <li>SelfExclusionFor5Years = 8,</li> <li>SelfExclusionFor7Years = 9,</li> <li>CoolOffFor24Hours = 10,</li> <li>CoolOffFor7Days = 11,</li> <li>CoolOffFor30Days = 12,</li> <li>CoolOffFor3Months = 13,</li> <li>CoolOffUntilSelectedDate = 14,</li> <li>CoolOffNone = 15</li> </ul>
expiryDate	string		Date when the self exclusion period will expire. Value is "Null" if account is not self-excluded.
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example (no self exclusion)

```

{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "period": 0,

```

```

"expiryDate": "2018-12-25 15:24:35",

"success": 1,

"timestamp": "2018-12-19 15:24:38",

"version": "1.0"
}

```

### g) GetCoolOffSettings

Checks if the user can see cool-off settings in front end.

URL: |GmServiceURL|/ServerAPI/GetCoolOffSettings/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
CoolOffAllow24HoursOption	boolean		True - Specifies whether the cool-off period of 24 hours is available in the front end for the player
CoolOffAllow30DaysOption	boolean		True - Specifies whether the cool-off period of 30 days is available in the front end for the player
CoolOffAllow3MonthsOption	boolean		True - Specifies whether the cool-off period of 3 Months is available in the front end for the player
CoolOffAllow7DaysOption	boolean		True - Specifies whether the cool-off period of 7 days is available in the front end for the player
CoolOffAllowUntilOption	boolean		True - Specifies whether the cool-off period until a selected date is available in the front end for the player
CoolOffEnableFrontEndOptions	boolean		True - Specifies whether the cool-off section is visible for the player
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```

{

  "CoolOffAllow24HoursOption": false,

  "CoolOffAllow30DaysOption": false,

  "CoolOffAllow3MonthsOption": false,

  "CoolOffAllow7DaysOption": false,

  "CoolOffAllowUntilOption": false,

  "CoolOffEnableFrontEndOptions": false,

  "errorData": {

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

  },

  "success": 1,

  "timestamp": "2018-08-02 13:42:30",

  "version": "1.0"
}

```



```
}
```

## h) GetSelfExclusionSettings

Checks if the user can see self-exclusion settings in the front end.

URL: |GmServiceURL|/ServerAPI/GetSelfExclusionSettings/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: GET

### Response variables

Name	Data type	Size	Comments
SelfExclusionAllow1YearOption	boolean		True - Specifies whether the self-exclusion period of 1 year is available in the front end for the player
SelfExclusionAllow5YearsOption	boolean		True - Specifies whether the self-exclusion period of 5 years is available in the front end for the player
SelfExclusionAllow6MonthsOption	boolean		True - Specifies whether the self-exclusion period of 6 months is available in the front end for the player
SelfExclusionAllowPermanentOption	boolean		True - Specifies whether the permanent self-exclusion period is available in the front end for the player
SelfExclusionAllowUntilOption	boolean		True - Specifies whether the self-exclusion period until a selected date is available in the front end for the player
SelfExclusionEnableFrontEndOptions	boolean		True - Specifies whether the self-exclusion section is visible for the player
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Response example

```
{
  "SelfExclusionAllow1YearOption": false,
  "SelfExclusionAllow5YearsOption": false,
  "SelfExclusionAllow6MonthsOption": false,
  "SelfExclusionAllowPermanentOption": false,
  "SelfExclusionAllowUntilOption": false,
  "SelfExclusionEnableFrontEndOptions": false,
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-02 14:02:04",
  "version": "1.0"
}
```

## i) SetUserRgCoolOff

Sets the user's cool-off period.

URL: |GmServiceURL|/ServerAPI/SetUserRgCoolOff/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

### Request variables

Name	M/O	Data type	Size	Comments
period	M	int		CoolOffFor24Hours = 10, CoolOffFor7Days = 11, CoolOffFor30Days = 12, CoolOffFor3Months = 13, CoolOffUntilSelectedDate = 14
sessionId	M	string		User's session ID. See method LoginUser for how to get SessionId
userId	O	long		The players' unique ID in GM.
Send Email Notification	O	boolean		Whether to send an email notification to the player informing him that he is being transferred to cool-off state
ExpiryDate	O	string		The date at which the cool-off period ends. This should only be set if the period parameter has the value 14 (CoolOffUntilSelectedDate), and should be a date between the current date and a maximum of 6 months from the current date. The value needs to be in the <b>JSON Serializer format</b> (milliseconds since the beginning of the Unix Epoch with a TimeZone) and a Epoch Converter needs to be used.
CoolOffDescription	O	string		The description of the cool off
CoolOffReason	O	string		The reason for the cool off
UnsatisfiedDescription	O	string		The description for the unsatisfaction
UnsatisfiedReason	O	string		The reason for the unsatisfaction

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, ErrorCode = 0, LogId = 0, ErrorDetails = "" (empty), ErrorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Request example

```
{  
  
  "CoolOffDescription": "test",  
  
  "CoolOffReason": "test",  
  
  "ExpiryDate": "VDate(928149600000+0000)V",  
}
```

```

"SendEmailNotification":true,

"UnsatisfiedDescription":"test",

"UnsatisfiedReason":"test",

"period":10,

"sessionId":"a1252478-5873-47e0-9d6e-fe00ce2e8ca2"

"userId": 3328116

}

```

#### Response example

```

{

  "errorData": {

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

  },

  "success": 1,

  "timestamp": "2018-08-07 13:27:18",

  "version": "1.0"

}

```

#### j) GetUserRgDepositLimitList

Returns the deposit limits set for the user.

URL: |GmServiceURL|/ServerAPI/GetUserRgDepositLimitList/ {VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
depositLimitData	object array		Deposit limit data
*amount	decimal		Deposit limit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*currency	string		Deposit limit currency
*period	int		0 - none, 1 - daily, 2 - weekly, 3 - monthly
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example

```

{

  "depositLimitData":[{

    "amount":12678967.54,

```

```

    "currency": "String content",

    "period": 2147483647

  }},

  "errorData": {

    "errorCode": 2147483647,

    "errorDetails": ["String content"],

    "errorMessage": "String content",

    "logId": 9223372036854775807

  },

  "success": 2147483647,

  "timestamp": "String content",

  "version": "String content"

}

```

### k) GetAvailableRealityCheckValues

This method is used to get available reality check values for a specified user.

URL: |GmServiceURL|/ServerAPI/GetAvailableRealityCheckValues/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Response variables

Name	Data type	Size	Comments
availableRealityCheckValues	string		This variable will return an empty list value if there is no reality check for the user's license or a set of values {0,30, 60, 90, 120} (those Values are for UK license, this values may vary for different license)
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request variables from URL

Name	M/O	Data type	Size	Comments
version	M		string	
partnerID	M		string	Unique ID for the operator
partnerKey	M		string	Unique Key for the operator
*sessionId	M		string	User session ID

#### Response example

```

{

  "availableRealityCheckValues": [

    "0",


```

```

    "30",
    "60",
    "90",
    "120",
    "180",
    "240",
    "300",
    "360",
    " (O_O)"
  ],
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 08:22:19",
  "version": "1.0"
}

```

## I) GetUserRealityCheck

This method is used to get the reality check for a user. This reality check was set earlier by the user.

URL: GmServiceURL/ServerAPI/GetUserRealityCheck/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

### Request variables from the URL

Name	M/O	Data type	Size	Comments
version	M		string	
partnerID	M		string	Unique ID for the operator
partnerKey	M		string	Unique Key for the operator
*sessionId	M		string	User session ID

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Error details
*errorMessage	string		Error message
*logId	int		Error log ID in the system
success	int	1	1 – success, 0 – failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

userRealityCheckValues	string	This variable will return an empty string if there is no reality check for the user or one of this values {30, 60, 90, 120} (those Values are for UK license, this values may vary for different license)
------------------------	--------	---

#### Response example

```
{
  "errorData":{
    "errorCode":2147483647,
    "errorDetails":["String content"],
    "errorMessage":"String content",
    "logId":9223372036854775807
  },
  "success":value,
  "timestamp":"content",
  "version":"content" ,
  "userRealityCheckValues":"content"
}
```

#### m) SetUserRealityCheck

This method is used to set for a specified user a reality check value.

URL: GmServerURL/ServerAPI/SetUserRealityCheck/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables from the URL

Name	M/O	Data type	Size	Comments
version	M	string		Version used, set to "1.0"
partnerID	M	string		Unique ID for the operator
partnerKey	M	string		Unique Key for the operator

#### Variables from the method (from the body)

Name	M/O	Data type	Size	Comments
sessionId	M	string		Session ID of the user
realityCheckValue	M	string		The value that you want to set as a reality check value, one of this values {0,30, 60, 90, 120} (those Values are for UK license, this values may vary for different license)

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Error details
*errorMessage	string		Error message
*logId	int		Error log ID in the system
success	int	1	1 – success, 0 – failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example (only for POST methods)

```
{
```

```

"realityCheckValue": "180",

"sessionId": "a1252478-5873-47e0-9d6e-fe00ce2e8ca2"

}

```

#### Response example

```

{

  "errorData": {

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

  },

  "success": 1,

  "timestamp": "2018-08-07 13:23:21",

  "version": "1.0"

}

```

#### n) CheckDepositTrans

Checks if deposit transaction can be processed, according to the player's responsible gambling settings for deposit limits. The method also runs all verifications as the Deposit method and afterwards, for the processing of the deposit, the Deposit ServerAPI method will run them once more.  
 URL: |GmServiceURL|/CheckDepositTrans/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

#### Request variables

Name	Data type			Comments
	M/O	Type	Size	
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	O	string		User session identifier
userId	O	string		The unique user id in GM
debitPayCardId	O	long		User's debit paycard identifier. If not provided, debitPayCardVendorId should be passed
debitPayCardVendorId	O	int		User's debit paycard vendor identifier. If not provided, debitPayCardId should be passed
creditAccountId	O	long		User's credit account identifier. Received from GetUserAccounts API.
requestAmount	M	decimal		Requested deposit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
requestCurrency	M	string		Requested deposit amount's currency
userIp	M	string		User IP (IPv4 format)
applyBonusVendorId	O	int		Bonus vendor identifier
applyBonusCode	O	string		Bonus code
isMobile	O	int		"1" if it is mobile, otherwise "0"
depositSource	O	int		Source of the tx: NotAvailable=0, Unknown=1, MainDepositPage=2, QuickDepositPage=3
iovationBlackBox	O	string		Device unique code

#### Response variables

Name	Data type	Comments
------	-----------	----------

	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

### Request example

http://localhost/ServerAPI/CheckDepositTrans/1.0/testID/testCode

```
{
  "userId": "23455322",
  "debitPayCardId": 0,
  "debitPayCardVendorId": 1063,
  "creditAccountId": 1524784,
  "requestAmount": 2,
  "requestCurrency": "EUR",
  "userIp": "127.0.0.1",
  "applyBonusVendorId": 0,
  "applyBonusCode": null,
  "depositSource": 0,
  "iovationBlackBox": null
}
```

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-04-27 17:27:59",
  "version": "1.0"
}
```

### o) SetUserRgLossLimit



This method is used in order for a player to set a Loss Limit on their gaming account, in order to establish a protective environment regarding the maximum amount the player can lose. It provides a way for the user to control himself and his losses.

URL: |GmServiceURL|/SetUserRgLossLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}/{AMOUNT}/{CURRENCY}

HTTP Method: GET

#### Response variables:

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether loss limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
  "success": 1,
  "timestamp": "2016-06-13 16:08:29",
  "version": "1.0"
}
```

#### p) SetUserRgWageringLimit

This method is used in order for a player to set a Wagering Limit on their gaming account, in order to establish a protective environment regarding the maximum amount the player can wager. It provides a way for the user to control himself and his losses. The Wagering Limit is different from the Loss Limit because wagering refers only to the amount played and it does not track if the bets/spins were won or lost.

URL: |GmServiceURL|/SetUserRgWageringLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}/{AMOUNT}/{CURRENCY}

HTTP Method: GET

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details

*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether loss limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
  "success": 1,
  "timestamp": "2016-06-13 16:08:29",
  "version": "1.0"
}
```

#### q) SetUserRgSessionLimit

This method is used in order for a player to set a Session Limit on their gaming account, in order to establish a protective environment regarding the maximum amount of time the player can spend in the system. It provides a way for the user to control himself and his losses. If the Session Limit is exceeded, the player will be logged out and will be able to resume playing only by logging in again.

URL: |GmServiceURL|/SetUserRgSessionLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{MINUTES}

HTTP Method: GET

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether loss limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Response example

```
{
```

```

"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
"isScheduled": 0,
"success": 1,
"timestamp": "2016-06-13 16:08:29",
"version": "1.0"
}

```

## r) GetUserRgLossLimit

This method is used by a back office user to request the active Loss Limit of a player, in order to use the information in the data flows and display it in the user interface.

URL: |GmServiceURL|/GetUserRgLossLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: GET

### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"
lossLimitData		object		Loss limit data
*amount		decimal		Loss limit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*currency		string		Loss limit currency
*period		int		Loss limit period: 0-none, 1-daily, 2-weekly, 3-monthly
*scheduled		object		Scheduled loss limit update data
*updateTime		string		Loss limit update time, yyyy-MM-dd HH:mm:ss

### Response example

```

{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },

```

```

"success": 1,
"timestamp": "2016-06-13 16:08:29",
"version": "1.0"
"lossLimitData": {
    "amount": 200,
    "currency": "EUR",
    "period": 1,
    "scheduled": null
}
}

```

#### Response example (with scheduled update)

```

{
    "errorData": {
        "errorCode": 0,
        "errorDetails": [],
        "errorMessage": "",
        "logId": 0
    },
    "success": 1,
    "timestamp": "2016-06-13 16:08:29",
    "version": "1.0"
    "lossLimitData": {
        "amount": 200,
        "currency": "EUR",
        "period": 1,
        "scheduled": {
            "amount": 480,
            "currency": "EUR",
            "period": 1,
            "updateTime": "2016-06-13 16:08:29",
        }
    }
}

```

#### s) GetUserRgWageringLimit

This method is used by a back office user to request the active Wagering Limit of a player, in order to use the information in the data flows and display it in the user interface.

URL: |GmServiceURL|/GetUserRgWageringLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

## Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"
wageringLimitData		object		Wagering limit data
*amount		decimal		Wagering limit amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*currency		string		Wagering limit currency
*period		int		Wagering limit period: 0-none, 1-daily, 2-weekly, 3-monthly
*scheduled		object		Scheduled wagering limit update data
*updateTime		string		Wagering limit update time, yyyy-MM-dd HH:mm:ss

## Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2016-06-13 16:08:29",
  "version": "1.0"
  "wageringLimitData": {
    "amount": 200,
    "currency": "EUR",
    "period": 1,
    "scheduled": null
  }
}
```

## Response example (with scheduled update)

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
```

```

        "errorMessage": "",
        "logId": 0
    },
    "success": 1,
    "timestamp": "2016-06-13 16:08:29",
    "version": "1.0"
    "wageringLimitData": {
        "amount": 200,
        "currency": "EUR",
        "period": 1,
        "scheduled": {
            "amount": 480,
            "currency": "EUR",
            "period": 1,
            "updateTime": "2016-06-13 16:08:29",
        }
    }
}

```

#### t) GetUserRgSessionLimit

This method is used by a back office user to request the Session Limit of a player, in order to use the information in the data flows and display it in the user interface.

URL: |GmServiceURL|/GetUserRgSessionLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}  
 HTTP Method: GET

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"
sessionLimitData		object		Session limit data
*minutes		int		Session limit period in minutes
*scheduled		object		Scheduled session limit update data
**minutes		int		Session limit update period in minutes
**updateTime		string		Session limit update time, yyyy-MM-dd HH:mm:ss

#### Response example

```

{

```

```

"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
},
"sessionLimitData": {
    "minutes": 25,
    "scheduled": null
},
"success": 1,
"timestamp": "2016-06-09 12:23:17",
"version": "1.0"
}

```

#### Response example (with scheduled update)

```

{
    "errorData": {
        "errorCode": 0,
        "errorDetails": [],
        "errorMessage": "",
        "logId": 0
    },
    "sessionLimitData": {
        "minutes": 25,
        "scheduled": {
            "minutes": 35,
            "updateTime": "2016-06-09 13:23:36"
        }
    },
    "success": 1,
    "timestamp": "2016-06-09 12:41:10",
    "version": "1.0"
}

```

#### u) RemoveUserRgWageringLimit

This method is used by a back office user to remove a wagering limit from a player's account, in order for the player to not be limited by responsible gambling anymore. For removing any type of wagering limit, a seven-day waiting period will be activated, which is meant to give the user time to consider the change.

URL: |GmServiceURL|/RemoveUserRgWageringLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}  
HTTP Method: GET

#### Request variables

Name	M/O	Data type		Comments
		Type	Size	
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	M	string		User's session identifier

#### Response variables

Name	M/O	Data type		Comments
		Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether loss limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

http://localhost/ServerAPI/RemoveUserRgWageringLimit/1.0/testID/testCode/4fe9c0c3-8ae6-4497-a724-03fa3f85eccf

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
  "success": 1,
  "timestamp": "2016-06-13 16:08:29",
  "version": "1.0"
}
```

#### v) RemoveUserRgSessionLimit



This method is used by a back office user to remove a session limit from a player's account, in order for the player to not be limited by responsible gambling anymore. For removing any type of session limit, a seven-day waiting period will be activated, which is meant to give the user time to consider the change.

URL: |GmServiceURL|/RemoveUserRgSessionLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}  
HTTP Method: GET

#### Request variables

Name	M/O	Data type		Comments
		Type	Size	
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	M	string		User's session identifier

#### Response variables

Name	M/O	Data type		Comments
		Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether loss limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

http://localhost/ServerAPI/RemoveUserRgSessionLimit/1.0/testID/testCode/4fe9c0c3-8ae6-4497-a724-03fa3f85eccf

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
  "success": 1,
  "timestamp": "2016-06-13 16:08:29",
  "version": "1.0"
```

```
}
```

### w) RemoveUserRgDepositLimit

This method is used by a back office user to remove a deposit limit from a player's account, in order for the player to not be limited by responsible gambling anymore. For removing any type of deposit limit, a seven-day waiting period will be activated, which is meant to give the user time to consider the change.

URL: |GmServiceURL|/RemoveUserRgDepositLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}  
HTTP Method: GET

#### Request variables

Name	Data type			Comments
	M/O	Type	Size	
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	M	string		User's session identifier
period	M	int		1 – daily, 2 – weekly, 3 – monthly

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether loss limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

http://localhost/ServerAPI/RemoveUserRgDepositLimit/1.0/testID/testCode/0e794c0c-eddb-4fc7-9e8b-5519363776c7/1

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
  "success": 1,
```

"timestamp": "2016-06-13 16:08:29",

"version": "1.0"

}

## x) RemoveUserRgLossLimit

This method is used by a back office user to remove a loss limit from a player's account, in order for the player to not be limited by responsible gambling anymore. For removing any type of loss limit, a seven-day waiting period will be activated, which is meant to give the user time to consider the change.

URL: |GmServiceURL|/RemoveUserRgLossLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

### Request variables

Name	M/O	Data type		Comments
		Type	Size	
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key
sessionId	M	string		User's session identifier

### Response variables

Name	M/O	Data type		Comments
		Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether loss limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

### Request example

http://localhost/ServerAPI/RemoveUserRgLossLimit/1.0/testID/testCode/4fe9c0c3-8ae6-4497-a724-03fa3f85eccf

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
```

```

    "success": 1,

    "timestamp": "2016-06-13 16:08:29",

    "version": "1.0"

}

```

### y)SetRgCumulativeSessionLimit

Sets a maximum time limit to a players account over a specified period. Unlike SetUserRgSessionLimit, the session timer does not reset upon a new login. If the limit is reached the player will have to wait for the pre-established time limit to reset. When a cumulative session limit is changed/created , the player will be logged out automatically - in session Agent (session terminated).

URL: {{url}}SetRgCumulativeSessionLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}/{MINUTES}  
 HTTP Method: GET

#### Request variables

Name	Data type	Size	Comments
sessionId	string		The user's session ID.
period	int	1	Time period over which the cumulative session limit applies. Possible values are:  1 - daily  2 - weekly  3 - monthly
minutes	int		Number of minutes per period until access is restricted to the player.

#### Request example:

```
{{url}}SetRgCumulativeSessionLimit/1/{{jetbull}}/d367cddd-859b-4309-aaea-1ad3fc0b55e1/1/350
```

#### Response variables:

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
isScheduled	int		Flag indicating whether deposit limit is already set. 1 - true, 0 - false
success	int	1	1 - success, 0 - failed
timeStamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Response example:

```

{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "isScheduled": 0,
  "success": 1,
  "timestamp": "2019-04-10 16:08:29",
  "version": "1.0"
}

```

### z)GetRgCumulativeSessionLimits

Retrieves the player's Responsible Gambling cumulative session limits settings.

URL: |GmServiceURL|/GetRgCumulativeSessionLimits/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

#### Request variables:

Name	M/O	Data type	Size	Comments
sessionId		String		The player's session ID.

#### Request example:

{{url}}GetRgCumulativeSessionLimit/{{jetbull}}/8ec1579a-4333-485b-bfb2-f04a817a2556

#### Response parameters:

Name	M/O	Data type	Size	Comments
cumulativeSessionLimitData		object		Contains details about the cumulative session limit
*minutes		int		Maximum number of minutes a player can use his account over specified period.
*period		int	1	Possible values are:  1 - Daily limit  2 - Weekly limit  3 - Monthly limit
*setDate		date		Returns date when the cumulative session limit was initially applied to the account in UNIX format.
*modifiedDate		date		Returns date when the cumulative session limit was last modified in UNIX format.
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Response example:

```
{
  "cumulativeSessionLimitData": [{
    "minutes": 355,
    "period": 1
    "setDate": "Date(1554631517443+0000)/"
    "modifiedDate": "Date(1554631955563+0000)/"
  }],
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 2036854775807
  },
  "success": 1,
  "timestamp": "2019-04-10 16:08:29",
  "version": "1.0"
}
```

## aa)RemoveRgCumulativeSessionLimit

Removes a player's cumulative session limit.

URL: |GmServiceURL|/RemoveRgCumulativeSessionLimit/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}/{PERIOD}

HTTP Method: GET

### Request variables

Name	Data Type	Size	Comments
sessionId	String		The player's session ID.
period	Int	1	Possible values are:  1 - Daily limit  2 - Weekly limit  3 - Monthly limit

### Request example:

```
{{url}}RemovetRgCumulativeSessionLimit/{{jetbull}}/8ec1579a-4333-485b-bfb2-f04a817a2556/1
```

### Response variables

Name		Data type	Comments	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		long		Error log ID in the system
isScheduled		int	1	1 - true, 0 - false  Flag, indicating whether cumulative session limit is applied or scheduled. If scheduled, it will be applied after predefined cooling period
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

### Response example:

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 07:56:12",
  "isScheduled": 0,
  "version": "1.0"
}
```

## 7. Partner Matrix 2

### a) AgentTransfer

Transfers money to/from an Agent account from/to a user's account. Permission is granted based on whether or not there is an agent relationship between the users. The user account must have the metadata key 'PM2AGENTID' set with the value being the ID of the agent who owns that user  
URL: |GmServiceURL|/AgentTransfer/{VERSION}/{PARTNERID}/{PARTNERKEY}  
HTTP Method: POST

#### Request variables

Name	M /O	Data type	Size	Comments
agentAccountID	M	long		The parent user Account ID
amount	M	decimal		The amount of money to be transferred. A maximum of 2 decimal places can be used after the integer for proper functionality.
currency	M	string		Currency
description	M	string		Comment added by agent
direction	M	string		Credit/Debit
userAccountID	M	long		The user account id
sessionID	M	string		The agent's session ID

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
transactionID	long		Id of the transaction
version	string		Version used, set to "1.0"

#### Request example

```
{  
  
  "agentAccountID":8261124,  
  
  "amount":1,  
  
  "currency":"EUR",  
  
  "description":"PM2 test",  
  
  "direction":"credit",  
  
  "sessionID":"38ec6441-cdc6-4a75-89d8-21e8cbd4e07f",  
  
  "userAccountID":8273038  
}
```

#### Response example

```
{  
  
  "errorData": {  
  
    "errorCode": 0,  
  

```

```

    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-08 07:56:12",
  "transactionID": 4982913,
  "version": "1.0"
}

```

## b) AgentImpersonateUser

The agent must be able to 'log in' to GM Core on behalf of his users, in order to place bets on his behalf.  
 URL: |GmServiceURL|/AgentImpersonateUser/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

### Request variables

Name	M/O	Data type	Size	Comments
agentSessionID	M	string		Agent session ID. See method LoginUser for how to get SessionId
userID	M	string		The user's ID for which the impersonating is needed

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" ((empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"
sessionId	string		User's session ID. See method LoginUser for how to get SessionId

### Request example

```

{
  "agentSessionID": "f3e6ceec661d4418afd7843a32ea0dd9",
  "userID": 3201254
}

```

### Response example

```

{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  }
}

```



```

},
"sessionId": "be62ccb5-65a1-4feb-abcb-5443e1ac74d7",
"success": 1,
"timestamp": "2018-08-06 13:06:20",
"version": "1.0"
}

```

### c) GetAccountBalanceForMultipleUsers

This ServerAPI method has been developed for PartnerMatrix 2, to be able to call the balances and their currencies for multiple users. This method is needed for agents to manage bets in players' name.

URL: |GmServiceURL|/GetAccountBalanceForMultipleUsers/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

#### Request variables

Name	Data type			Comments
	M/O	Type	Size	
version	M	string		API version, set to "1.0"
partnerID	M	string		API partner identifier
partnerKey	M	string		API partner key

#### PayLoad

Name	Data type			Comments
	M/O	Type	Size	
sessionId	M	String	32	Session Id of the Agent
userIds	M	List<long>		Ids of users to query

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = [] (empty), errorMessage = "" (empty)
errorCode		int		Error code
errorDetails		string		Possible error details
errorMessage		string		Error message
logId		long		Error log ID in the system
accountsBalance		object		List of Accounts Balances
balanceAmount		decimal		The user's balance amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
currency		String		Currency
userID		long		User's Id
isBalanceAvailable		bool		0 if balance not available, 1 if balance is available
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

<http://dev.gammatrix.com/ServerAPI/GetAccountBalanceForMultipleUsers/1.0/floortestID/floortestCode>

```
{
  "sessionId": "21d92028-0f37-4283-8beb-1c8d2c34c968",
  "userIds": [2984148, 2984147]
}
```

#### Response example

```
{
  "accountsBalance": [
    {
      "balanceAmount": 0,
      "currency": "EUR",
      "userID": 2984148,
      "isBalanceAvailable": true
    },
    {
      "balanceAmount": 0,
      "currency": "EUR",
      "userID": 2984147,
      "isBalanceAvailable": true
    }
  ],
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2016-10-24 15:33:29",
  "version": "1.0"
}
```

#### d) AgentResetPlayerPassword

This ServerAPI method has been developed for PartnerMatrix 2, to allow an agent to reset password for managed players.

URL: |GmServiceURL|/AgentResetPlayerPassword/{VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

#### Request variables

Name	Data type	Comments
------	-----------	----------

	M/O	Type	Size	
agentSessionId	M	string		Session token for the Agent performing the operation (usually a guid)
userId	M	Int 64		ID of the player whose password is being reset
newPassword	M	string		New plain text password
confirmNewPassword	M	String		New plain text password, must be identical to newPassword

#### Request example

```
{
  "agentSessionId": "a684f71c-e6c1-46ed-9763-9c5e310c08e4",
  "confirmNewPassword": " newPassForUser ",
  "newPassword": " newPassForUser",
  "userId": 3000146
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2017-05-30 16:41:14",
  "version": "1.0"
}
```

## 8. Monitoring

### a) DomainHealthCheck

This method returns the status of the providers specific to a domain.

URL: |GmServiceURL/ServerApi/DomainHealthCheck/version/siteID/siteCode/includeDetails

HTTP Method: POST

#### Request variables

Name	M/O	Type	Size	Comments
siteID	M	string		Assigned site ID, provided by Integration manager.
siteCode	M	string		Assigned site code, provided by Integration manager.
includeDetails	O	String		Values: true/false.
Version	M	string		Version used, set to "1.0"

#### Response variables

Name	M/O	Type	Size	Comments
------	-----	------	------	----------

Status		string	Possible values: <i>None = 0, Up = 1, PartiallyUp = 2, Down = 3,</i>
Details		object	Object presenting details for each vendor
*Method		string	Gaming vendor API name
*Vendor		string	Gaming vendor name
*Enabled		boolean	Status of the API method. Possible values: <i>"true", "false"</i>
*Importance		string	Importance of API method. Possible values: <i>None = 0, Critical = 1, NonCritical = 2</i>
errorData		object	If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode		int	Error code
*errorDetails		string	Possible error details
*errorMessage		string	Error message
*logId		int	Error log ID in the system
Timestamp		datetime	Date and time of the request
Version		string	Version used, set to "1.0"

#### Request example with active providers

<https://core-gm-stage.everymatrix.com/ServerAPI/DomainHealthCheck/1.0/testID/testCode/true>

#### Response example with active providers

```
{
  "details":[
    {
      "Enabled":true,
      "Importance":1,
      "Method":"PING",
      "Vendor":"NetEnt"
    },
    {
      "Enabled":true,
      "Importance":1,
      "Method":"REGISTER",
      "Vendor":"NetEnt"
    },
    {
      "Enabled":true,
      "Importance":1,
      "Method":"LOGIN",
      "Vendor":"NetEnt"
    },
    {
      "Enabled":true,
      "Importance":1,
      "Method":"GETBALANCE",
```

```
"Vendor": "NetEnt"
},
{
  "Enabled": true,
  "Importance": 1,
  "Method": "FUNDSTRANSFER",
  "Vendor": "NetEnt"
},
{
  "Enabled": true,
  "Importance": 1,
  "Method": "BONUS",
  "Vendor": "NetEnt"
},
{
  "Enabled": true,
  "Importance": 1,
  "Method": "REALTIMEAPI",
  "Vendor": "NetEnt"
},
{
  "Enabled": true,
  "Importance": 1,
  "Method": "REALTIMEAPI",
  "Vendor": "IGT"
},
{
  "Enabled": true,
  "Importance": 1,
  "Method": "PERIODICFEED",
  "Vendor": "NetEnt"
}
],
"errorData": {
  "errorCode": 0,
  "errorDetails": [],
  "errorMessage": "",
  "logId": 0
}
```

```

},
"status":1,
"success":1,
"timestamp":"2017-09-25 14:12:56",
"version":"1.0"
}

```

#### Request example with no providers active

https://core-gm-stage.everymatrix.com/ServerAPI/DomainHealthCheck/1.0/testID/testCode/false

#### Response example with no providers active

```

{
  "details":null,
  "errorData":{
    "errorCode":0,
    "errorDetails":[],
    "errorMessage":"",
    "logId":0
  },
  "status":1,
  "success":1,
  "timestamp":"2017-09-25 14:13:47",
  "version":"1.0"
}

```

## b)GlobalHealthCheck

This method returns the status of the GamMatrix Core external dependencies which are relevant for all the domains hosted on a GamMatrix Core environment.

It checks database status, session agent status, currency rates status and IP database status.

URL: |GmServiceURL/ServerApi/GlobalHealthCheck/version/siteID/siteCode

HTTP Method: GET

#### Request variables

Name	M/O	Type	Size	Comments
siteID	M	string		Assigned site ID, configured on System domain. Setting is stored in AdminTool (internal EM tool), under the path: <b>Services &gt; ServerAPI</b>
siteCode	M	string		Assigned site code, configured on System domain. Setting is stored in AdminTool (internal EM tool), under the path: <b>Services &gt; ServerAPI</b>
version	M	string		Version used, set to "1.0"

#### Response variables

Name	M/O	Type	Size	Comments
Status	M	string		Possible values:  <i>None = 0,</i>  <i>Up = 1,</i>  <i>PartiallyUp = 2,</i>  <i>Down = 3,</i>
Details	M	object		Object presenting details for each vendor
*Vendor	M	string		Gaming vendor name
*Enabled	M	boolean		Status of the API method. Possible values: <i>"true"</i> , <i>"false"</i>
*Importance	M	string		Importance of API method. Possible values: <i>None = 0, Critical = 1, NonCritical = 2</i>
errorData	M	object		If success,  errorCode = 0,  logId = 0,  errorDetails = "" (empty),  errorMessage = "" (empty)
*errorCode	M	int		Error code
*errorDetails	M	string		Possible error details
*errorMessage	M	string		Error message
*logId	M	int		Error log ID in the system
Timestamp	M	datetime		Date and time of the request
Version	M	string		Version used, set to "1.0"

#### Request example

<https://core-gm-stage.everymatrix.com/ServerApi/GlobalHealthCheck/1.0/testID/testCode>

#### Response example

```
{
  "details":{
    "Details":[
      {
        "Enabled":true,
        "Importance":1,
        "Vendor":"GmCore DB"
      },
      {
        "Enabled":true,
        "Importance":1,
        "Vendor":"Cm DB"
      },
      {
        "Enabled":true,
        "Importance":1,
```

```

        "Vendor": "Logs DB"
    },
    {
        "Enabled": true,
        "Importance": 1,
        "Vendor": "SessionAgent"
    },
    {
        "Enabled": true,
        "Importance": 2,
        "Vendor": "CurrencyRatesUpdate"
    },
    {
        "Enabled": true,
        "Importance": 2,
        "Vendor": "IpDatabaseUpdate"
    }
],
"Status": 1
},
"errorData": {
    "errorCode": 0,
    "errorDetails": [ ],
    "errorMessage": "",
    "logId": 0
},
"success": 1,
"timestamp": "2018-08-21 11:39:36",
"version": "1.0"
}

```

## 9. Linked Players Methods

### GetUserLinkedPlayers

Checks and returns all players related to a specific player in Linked Players feature according to the sent parameters

URL: /ServerAPI/GetUserLinkedPlayers/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

Request parameters

Name	M/O	Data type	Size	Comments
userId	O	Long		The player's user ID in GM.



personalId	O	Long		The player's personal ID.  If less than 5 characters, parameter will be ignored and considered non-valid.
------------	---	------	--	---

**Notes:**

If both parameters are present in the request and not valid, response will contain an error message.

If both parameters are present and valid in the request than only precise matches are returned.

Swedish License: If user has regCountry = SE and personalId matches, then the players will be linked as trusted.

Response parameters:

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	Long		Error log ID in the system
success	int	1	1 - Success, 0 - Failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"
userLinksList	array		List of players matching the request parameters.
activeStatus	int	1	1 – Active account. 0 - Inactive account
domainId	int		The domain ID to which the player registered
userId	Long		The player's user ID in GM.

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2019-02-12 11:58:14",
  "userLinksList": [
    {
      "activeStatus": 0,
      "domainId": 1008,
      "userId": 3327640
    },
    {
      "activeStatus": 0,
      "domainId": 1008,
      "userId": 3327481
    }
  ],
  "version": "1.0"
}
```

## Deprecated Methods

### Bonus

#### a) ApplyUserBonus(DEPRECATED)

Instruct the GamMatrix system to give bonus to a user.

URL: |GmServiceURL|/ServerAPI/ApplyUserBonus/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M /O	Data type	Size	Comments
accountId	M	long		User's gaming account ID
applyBonusCode	M	string		Bonus code
applyBonusVendorId	M	int		Gaming vendor ID. Always set to 112(CasinoWallet)
sessionId	M	string		User's session ID. See method LoginUser for how to get SessionId
iovationBlackBox	O	string	~2000 characters	The set of characters generated by the Iovation script embedded in the web page. This unique code corresponds to a certain device

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```
{  
  
  "accountId":6023282,  
  
  "applyBonusCode":"dd_nodepo1",  
  
  "applyBonusVendorId":112,  
  
  "iovationBlackBox":"04003hQUMXGB0poNf94lis1ztu1p0Y9k4RDY0Hfp5ie720zxMSrvHBQm916St",  
  
  "sessionId":"86404581-a1c6-4d7f-9b1a-bf2427013f0c"  
}
```

#### Request example with Iovation enabled

```
{  
  
  "iovationBlackBox":"04003hQUMXGB0poNf94lis1ztu1p0Y9k4RDY0Hfp5ie720zxMSrvHBQm916St",  
  
}
```

#### Response example

```
{
```

```

"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
"success": 1,
"timestamp": "2018-08-08 07:59:18",
"version": "1.0"
}

```

## b) GetUserBonusDetails(DEPRECATED)

Returns a list of active bonuses for the user.

URL: |GmServiceURL|/ServerAPI/GetUserBonusDetails/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}

HTTP Method: GET

### Response variables

Name	Data type	Size	Comments
bonusList	array		List of available bonuses for the user
*amount	decimal		Original bonus amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*bonusId	string		Bonus ID
*confiscateAllFundsOnExpiration	boolean		Confiscate all funds on expiration i.e. including deposit amount
*confiscateAllFundsOnForfeiture	boolean		Confiscate all funds on forfeit i.e. including deposit amount
*created	string		Date when bonus was issued. Format: yyyy-MM-dd hh:MM:ss
*currency	string		user account currency. See Appendix 2. Currency codes
*expiryDate	string		Bonus expiration date. Format: yyyy-MM-dd hh:MM:ss
*id	string		Internal bonus ID for the bonus
*name	string		Bonus name
*pointsLeftToCollect	decimal		
*pointsToCollect	decimal		
*remainingAmount	decimal		Current bonus amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*remainingWagerRequirementAmount	decimal		Remaining wagerin requirements amount. A maximum of 2 decimal places can be used after the integer for proper functionality.
*remainingWagerRequirementCurrency	string		Remaining wagering requirement currency
*status	string		0 - granted, 1 - useup, 2 - forfeited, 3 - expired, 4 - cashout
*type	string		Bonus type
*vendorId	int		Gaming vendor ID in the GamMatrix system
*wagerRequirementAmount	decimal		Original wagering requirements amount
*wagerRequirementCurrency	string		Original wagering requirement currency
*excludedGamesList	array		List of excluded game list
**key	string		Key representing Vendor ID
**value	array		List of strings, representing Game Codes
*playerBase	string		Information about player
**affCodes	array		Casino bonus level type - affiliate code

***excluded	boolean		true - casino bonus level type flag is EXCLUDED, false - casino bonus level type flag is INCLUDED
***value	string		
**affUsers	array		Casino bonus level type - PM affiliate user
***excluded	boolean		true - casino bonus level type flag is EXCLUDED, false - casino bonus level type flag is INCLUDED
***value	string		
**bonusIds	array		Casino bonus level type - bonus ID
***excluded	boolean		true - casino bonus level type flag is EXCLUDED, false - casino bonus level type flag is INCLUDED
***value	string		
**bonusTypes	array		Casino bonus level type - bonus type
***excluded	boolean		true - casino bonus level type flag is EXCLUDED, false - casino bonus level type flag is INCLUDED
***value	string		
**countries	array		Casino bonus level type - Country
***excluded	boolean		true - casino bonus level type flag is EXCLUDED, false - casino bonus level type flag is INCLUDED
***value	string		
**roles	array		Casino bonus level type - Role
***excluded	boolean		true - casino bonus level type flag is EXCLUDED, false - casino bonus level type flag is INCLUDED
***value	string		
**users	array		Casino bonus level type - User
***excluded	boolean		true - casino bonus level type flag is EXCLUDED, false - casino bonus level type flag is INCLUDED
***value	string		
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Response example

```
{
  "bonusList": [
    {
      "ExcludedGamesList": [],
      "PlayerBase": {
        "AffCodes": {
          "Excluded": false,
          "Value": ""
        },
        "AffUsers": {
          "Excluded": false,
          "Value": ""
        },
        "BonusIds": {
```

```
    "Excluded": false,
    "Value": ""
  },
  "BonusTypes": {
    "Excluded": false,
    "Value": ""
  },
  "Countries": {
    "Excluded": false,
    "Value": ""
  },
  "Roles": {
    "Excluded": false,
    "Value": ""
  },
  "Users": {
    "Excluded": false,
    "Value": "3201254"
  }
},
"amount": 50,
"bonusId": "1016364",
"confiscateAllFundsOnExpiration": false,
"confiscateAllFundsOnForfeiture": false,
"created": "2018-08-08 08:43:34",
"currency": "EUR",
"expiryDate": "2018-09-22 08:43:34",
"id": "5599081",
"name": "ndbnewest",
"pointsLeftToCollect": 0,
"pointsToCollect": 0,
"remainingAmount": 50,
"remainingWagerRequirementAmount": 100,
"remainingWagerRequirementCurrency": "EUR",
"status": "Granted",
"type": "No deposit",
"vendorId": 155,
"wagerRequirementAmount": 100,
```

```

        "wagerRequirementCurrency": "EUR"
    }
},
"errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
},
"success": 1,
"timestamp": "2018-08-08 08:43:59",
"version": "1.0"
}

```

### c) ForfeitUserCasinoBonus(DEPRECATED)

Forfeits a Casino bonus.

URL: |GmServiceURL|/ServerAPI/ForfeitUserCasinoBonus/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
accountId	M	long	1	User's Casino AccountId. See method GetUserAccounts
sessionId	M	string		User's session ID. See method LoginUser for how to get itSessionId
userCasinoBonusID	M	long		User's casino bonus ID

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```

{
    "accountId":234567,
    "sessionId":"3fb155db-6f11-4cd7-a53e-b23b297dff4c",
    "userCasinoBonusID":10287
}

```

#### Response example

```

{
    "errorData": {

```

```

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

  },

  "success": 1,

  "timestamp": "2018-08-08 08:16:32",

  "version": "1.0"

}

```

#### d) ApplyAffiliateChipBonus(DEPRECATED)

Applies the chip bonus for the affiliate.

URL: |GmServiceURL|/ServerAPI/ApplyAffiliateChipBonus /{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
applyToUserId	M	long		User ID
bonusCode	M	string		Bonus Code
bonusVendorId	M	int		Gaming Vendor ID
chipsAmount	M	int		Chips amount in the gaming account

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```

{

  "applyToUserId":3201254,

  "bonusCode":"test_chip_bonus",

  "bonusVendorId":112,

  "chipsAmount":6

}

```

#### Response example

```

{

  "errorData": {

    "errorCode": 0,

    "errorDetails": [],


```

```

    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-06 13:20:01",
  "version": "1.0"
}

```

### e) ApplyUserExternalBonus(DEPRECATED)

Instructs the GamMatrix system to give bonus to a user. Unlike the ApplyUserBonus method, which applies only NoDeposit bonus types, the ApplyUserExternalBonus also applies Deposit and Reload bonuses to the user. Deposit and bonus sums can be specified for the call, where needed.  
 URL: |GmServiceURL|/ApplyUserExternalBonus/ {VERSION}/{PARTNERID}/{PARTNERKEY}  
 HTTP Method: POST

#### Request variables (only for POST methods)

Name	M /O	Data type	Size	Comments
accountId	M	long		User's gaming account ID
applyBonusCode	M	string		Bonus code
applyBonusVendorId	M	int		Gaming vendor ID. Always set to 112(CasinoWallet) or 155(ExternalWallet). Can be taken from the user gaming account.
sessionId	M	string		User's session ID. See method LoginUser for how to get SessionId
bonusAmount	O	decimal		Amount applied as a bonus sum. For some bonus types, the amount is not fixed.
bonusCurrency	O	string		Currency for the bonus amount. Should be filled if the BonusAmount parameter is filled.
depositAmount	O	decimal		Amount applied as a bonus sum. For some bonus types, the amount is not fixed. A maximum of 2 decimal places can be used after the integer for proper functionality.
depositCurrency	O	string		Currency for the deposit amount. Should be filled if the DepositCurrency parameter is filled.
iovationBlackBox	O	string	~2000 characters	The set of characters generated by the Iovation script embedded in the web page. This unique code corresponds to a certain device

#### Response variables

Name	M/O	Data type	Size	Comments
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = ""(empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Error details
*errorMessage		string		Error message
*eogId		int		Error log ID in the system
success		int		1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used

#### Request example

```

{
  "accountId":67847839,
  "applyBonusCode":"DEP_BONUS_1",
  "applyBonusVendorId":155,
  "sessionId":"86404581-a1c6-4d7f-9b1a-bf2427013f0c",

```



```

"bonusAmount":100,

"bonusCurrency":"EUR",

"depositAmount":150,

"depositCurrency":"EUR"

}

```

#### Request example with iovation enabled

```

{

    "iovationBlackBox":"aehfzAktEupVh3mmOMPklfeZyvNPcNwxmxXMcS8KJZHhCXY6H2W0d",

}

```

#### Response example

```

{

    "errorData": {

        "errorCode": 0,

        "errorDetails": [],

        "errorMessage": "",

        "logId": 0

    },

    "success": 1,

    "timestamp": "2018-08-08 08:01:14",

    "version": "1.0"

}

```

#### f) GetAvailableBonusList(DEPRECATED)

Retrieves available bonus list.

URL: |GmServiceURL|/GetAvailableBonusList/{VERSION}/{PARTNERID}/{PARTNERKEY}/{SESSIONID}  
 HTTP Method: GET

#### Request variables

Name	M/O	Data type	Size	Comments
sessionId	M	string		User's session ID. See method LoginUser for how to get it

#### Response variables

Name	M/O	Data type	Size	Comments
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		string		Error log ID in the system
success		int	1	1 - success, 0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

bonusList		list(object)	List on bonuses
*vendorId		int	ID of the vendor
*bonusId		long	ID of the bonus
*name		string	Name of the bonus
*type		int	Type of the bonus
*amount		decimal	Amount of the bonus. A maximum of 2 decimal places can be used after the integer for proper functionality.
*currency		string	Currency used at bonus
*created		string	Date and time when it was created
*expiryDate		string	Date and time when it will expire
*wageringCoefficient		decimal	The coefficient of the wagering
*code		string	Code of the bonus

### Response example

```
{
  "bonusList": [
    {
      "amount": 50,
      "bonusId": 1016364,
      "code": "ndbnew1",
      "created": "2018-08-03 13:52:20",
      "currency": "EUR",
      "expiryDate": "2019-12-31 00:00:00",
      "name": "ndbnewest",
      "type": 1,
      "vendorId": 155,
      "wageringCoefficient": 2
    },
    {
      "amount": 22,
      "bonusId": 1015629,
      "code": "dmFDB",
      "created": "2016-12-22 07:44:19",
      "currency": "EUR",
      "expiryDate": "2017-01-22 00:00:00",
      "name": "dmFDB",
      "type": 0,
      "vendorId": 112,
      "wageringCoefficient": 1
    }
  ]
}
```

```

        "bonusId": 1003847,

        "code": "ABC123",

        "created": "2014-12-29 11:02:50",

        "currency": "EUR",

        "expiryDate": "2014-01-30 00:00:00",

        "name": "test",

        "type": 0,

        "vendorId": 112,

        "wageringCoefficient": 2

    }

],

"errorData": {

    "errorCode": 0,

    "errorDetails": [],

    "errorMessage": "",

    "logId": 0

},

"success": 1,

"timestamp": "2018-08-08 08:18:44",

"version": "1.0"

}

```

#### g) ApplicableBonusList(DEPRECATED)

This method is used in order to obtain the list with all eligible bonuses a specific user can take.

URL: |GmServiceURL|/ApplicableBonusList/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	Data type			Comments
	M/O	Type	Size	
accountId	M	int 64		ID of player account, as obtained from calling GetUserAccounts method
sessionId	M	string		Session token (usually a guid)
transType	M	int 32		Represents a transaction type. For the moment, only 0 is supported by this method (Deposit)

#### Response variables

Name	Data type			Comments
	M/O	Type	Size	
vendorId		int		VendorID of wallet
name		string		Name of bonus

type		int		Casino bonus type. Can be one of the values:  FirstDeposit = 0,  ReloadBonus = 2,  StandardBonus = 4.
code		string		Bonus code
description		string		Bonus description
errorData		object		If success,  errorCode = 0,  logId = 0,  errorDetails = "" (empty),  errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		string		Error log ID in the system
success		int	1	1 - success  0 - failed
timestamp		string		yyyy-MM-dd HH:mm:ss
version		string		Version used, set to "1.0"

#### Request example

```
{
  "accountId":8489991,
  "sessionId":"0bbc75e5-90f1-4662-ad6c-bbacf410baa5",
  "transType":0
}
```

#### Response example

```
{
  "bonusList": [
    {
      "code": "dmTest100",
      "description": "dmTest100",
      "name": "dmTest100",
      "type": 4,
      "vendorId": 112
    }
  ],
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": ""
  }
}
```

```

    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-06 13:17:57",
  "version": "1.0"
}

```

## h)SetUserPromotion(DEPRECATED)

This method saves a player's promotion preferences.

URL: |GmServiceURL|/SetUserPromotion/{version}/{partnerID}/{partnerKey}

HTTP Method: POST

### Request variables

Name	M/O	Type	Size	Comments
userid	M	long		The player's user id
promotionType	M	int		Promotion: 0, Bonus: 1
code	M	string		The promotion code identifier
title	O	string		Promotion description
actionType	M	int		Add: 0, Remove: 1

### Response variables

Name	M/O	Type	Size	Comments
Success		int		Can be either: <ul style="list-style-type: none"> <li>· 1 – success</li> <li>· 0 - failed</li> </ul>
errorData		object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message
*logId		int		Error log ID in the system
Timestamp	M	datetime		Date and time of the request
Version	M	string		Version used, set to "1.0"

### Request example

```

{
  "userId":3001121,
  "code":"PROMO2",
  "title":"Promo1 description",
  "promotionType":"Promotion",
  "actionType":"Add"
}

```

```
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-03-28 15:19:07",
  "version": "1.0"
}
```

#### i)GetUserPromotions(DEPRECATED)

This method retrieves a player's promotion preferences.

URL: |GmServiceURL|/GetUserPromotions/{version}/{partnerID}/{partnerKey}

HTTP Method: POST

#### Request variables

Name	M/O	Type	Size	Comments
userid	M	long		The player's user id
promotionType	M	long		Promotion details

#### Response variables

Name	M/O	Type	Size	Comments
userPromotionsList		object		Object describing the promotion details
*code		string		Promotion code
* ins		string		Date of promotion assignment
* promotionType		long		Promotion details
* title		string		Promotion details
* userID		long		The player id
Success		Int		Can be either: <ul style="list-style-type: none"><li>· 1 – success</li><li>· 0 - failed</li></ul>
errorData		object		If success,  errorCode = 0,  logId = 0,  errorDetails = "" (empty),  errorMessage = "" (empty)
*errorCode		int		Error code
*errorDetails		string		Possible error details
*errorMessage		string		Error message

*logId		int		Error log ID in the system
Timestamp		datetime		Date and time of the request
Version		string		Version used, set to "1.0"

#### Request example

```
{
  "userId":3001121,
  "promotionType":"Promotion",
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-03-28 15:12:15",
  "userPromotionsList": [
    {
      "code": "dmPROMo99aaa",
      "ins": "2017-10-20 07:56:49",
      "promotionType": "Promotion",
      "title": "PROMO2 description",
      "userId": 3001121
    }
  ],
  "version": "1.0"
}
```

## Transactions

### a) ManualDeposit (OBSOLETE)

Makes a manual deposit.

URL: |GmServiceURL|ServerAPI/ManualDeposit/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
amount	M	decimal		Amount to be deposited. A maximum of 2 decimal places can be used after the integer for proper functionality.
currency	M	string		Currency of the manual deposit
fromAccountId	M	long		Credit account id

sessionId	M	string	40	User's session ID. See method LoginUser for how to get SessionId
toAccountId	M	long		Debit account ID

#### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

#### Request example

```
{
  "amount":10,
  "currency":"EUR",
  "fromAccountId":215487,
  "sessionId":"be62ccb5-65a1-4feb-abcb-5443e1ac74d7",
  "toAccountId":8363737
}
```

#### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-06 13:35:05",
  "version": "1.0"
}
```

#### b) ManualWithdraw (OBSOLETE)

Makes a manual withdrawal.

URL: |GmServiceURL|ServerAPI/ManualWithdraw/{VERSION}/{PARTNERID}/{PARTNERKEY}

HTTP Method: POST

#### Request variables

Name	M/O	Data type	Size	Comments
amount	M	decimal		Amount to be deposited
currency	M	string		Currency of the manual deposit
fromAccountId	M	long		Credit account id



sessionId	M	string	40	User's session ID. See method LoginUser for how to get SessionId
toAccountId	M	long		Debit account id

### Response variables

Name	Data type	Size	Comments
errorData	object		If success, errorCode = 0, logId = 0, errorDetails = "" (empty), errorMessage = "" (empty)
*errorCode	int		Error code
*errorDetails	string		Possible error details
*errorMessage	string		Error message
*logId	long		Error log ID in the system
success	int	1	1 - success, 0 - failed
timestamp	string		yyyy-MM-dd HH:mm:ss
version	string		Version used, set to "1.0"

### Request example

```
{
  "amount":50,
  "currency":"EUR",
  "fromAccountId":8363737,
  "sessionId":"be62ccb5-65a1-4feb-abcb-5443e1ac74d7",
  "toAccountId": 215487
}
```

### Response example

```
{
  "errorData": {
    "errorCode": 0,
    "errorDetails": [],
    "errorMessage": "",
    "logId": 0
  },
  "success": 1,
  "timestamp": "2018-08-06 13:39:17",
  "version": "1.0"
}
```

## Appendix 1. Error codes

ErrorCode	ErrorMessage
"0"	No error - successful call
"1"	"Player doesn't have enough funds to withdraw this amount"
"2"	"No player with the sent username found"
"3"	"Player record already exists"
"5"	"Obligatory field missing"

"6"	"Invalid operator key"
"8"	"Banned player cannot deposit funds"
"9"	"There're more than one transaction with same ID"
"10"	"Transfer amount illegal (probably negative)"
"11"	"Error registering the transaction on live casino"
"13"	"The transaction exists and rejected on live casino"
"15"	"Player update failed"
"16"	"Player with the same email already exists"
"18"	"The transaction exists but with different details"
"19"	"Unspecified error"
"20"	"Loss limit exceeded"
"21"	"Session limit exceeded"
"22"	"Wagering limit exceeded"
"100"	"Unknown error"
"101"	"Invalid GameID"
"1000"	System error
"1001"	Duplicated email
"1002"	Username in use
"1003"	User authentication failed
"1004"	Invalid or expired session
"1005"	User account not found
"1006"	User account is blocked
"1007"	User account is not active
"1008"	User game account not found
"1009"	User game account is blocked
"1010"	User game account is not active
"1012"	User paycard is blocked
"1013"	User paycard is not active
"1020"	User casino wallet account not found
"1021"	User bonus not found
"1022"	PM2 invalid agent-user relationship
"1023"	PM2 invalid direction
"1024"	PM2 agent transfer not allowed
"1025"	Invalid vendor
"1026"	Bonus invalid direction
"1027"	User account is already active
"1028"	Invalid or expire hash
"1029"	Transaction not found
"10011"	User paycard not found
"10014"	Insufficient funds

"1030"	TransactionInProgress
"1031"	No deposit limit.
"1032"	Deposit limit reached.

## Appendix 2. Currency codes

Following currencies are currently supported (ISO 4217):

Code	Description	
EUR	Euro	
USD	US Dollar	
GBP	Pound Sterling	
CNY	Chinese Yuan	
CZK	Czech Koruna	
DKK	Danish Krone	
NOK	Norwegian Krone	
PLN	Polish Zloty	
RUB	Russian Ruble	
SEK	Swedish Krona	
ZAR	South African Rand	
CAD	Canadian Dollar	
AUD	Australian Dollar	
BGN	Bulgarian Lev	
GEL	Georgian Lari	
MXN	Mexican Peso	
NGC	Nigerian Naira	
RON	Romanian Leu	
TRY	Turkish Lira	
VEF	Venezuelan Bolivar	
HRK	Croatia Krune	
HUF	Hungarian Florint	
PEN	Peruvian Nuevo Sol	

## Appendix 3. Country codes

Following countries are currently supported (ISO 3166-1): <https://www.iso.org/obp/ui/#search>

## Appendix 4. Phone country codes

A list with the countries phone codes for may be found here: <http://countrycode.org/>

## Appendix 5. Language codes

ISO 639-1, two letter code. A list with the language codes may be found here: <http://www.mathguide.de/info/tools/languagecode.html>