Wallet Integration

Introduction

This document describes integration between *Integration Module* and an *Wallet* using seamless wallet adapter.

The key words `MUST`, `MUST NOT`, `REQUIRED`, `SHALL`, `SHALL NOT`, `SHOULD`, `SHOULD NOT`, `RECOMMENDED`, `MAY`, and `OPTIONAL` in this document are to be interpreted as described in RFC 2119.

Changelog

Version	Date	Changes
`1.0.5`	05.07.2020	• Removed `wallet` param
`1.0.4`	23.06.2020	• Separated *Wallet* and *Operator* • Added `provider` param to the launcher
`1.0.3`	19.06.2020	• Added `jackpotAmount`, `promotionType` and `promotionId` to `/transaction` request • Added Launcher section
`1.0.2`	18.06.2020	• Authorization header is now `Bearer <token>` instead of `<token>` • Removed `USER_NOT_FOUND` and `SESSION_NOT_FOUND` in favour of `USER_UNAUTHORIZED` • Removed `nativeID` from params • `Proxy-Authorisation` header changed to `X-Server-Authorization`
`1.0.1`	15.06.2020	• Added Automated Wallet Verifier section
`1.0.0`	01.06.2020	• First version

Set up information

The *Operator* `SHOULD` provide:

- Endpoints for staging and production environments of Operator.
- *Operator* name.
- Minimum and maximum bet, maximum exposure (all values in `eur`).

The *Operator* `SHOULD` be provided with:

- Endpoints for staging and production environments.
- IPs of production environment to whitelist.
- `SSL` public key.
- `HMAC` secret key.

Integration

Prerequisites:

- Communication `SHOULD` be done via `HTTP/1.1` and encrypted with `SSL/TLS`.

- All API requests `MUST` be passed with `Content-Type: application/json` .
- All API responses `MUST` be passed with `Content-Type: application/json` .
- Any additional response or request data, parameters, fields or codes not described in documentation `SHOULD NOT` be used.
- Character encoding utf-8 `SHOULD` be set to `UTF-8`
- All time and date fields `SHOULD` be in `UTC` timezone

Authorization

Authorization `MUST` be done using the following headers:

- `Authorization` : *User's* authorization with `Bearer <token>` where `<token>` comes from `/authentication` request. It `MUST` be used to identify a *User*.
- `X-Server-Authorization` : *Server's* authorization with `HMAC` scheme using `sha256` algorithm and request payload. Example of generating header: `crypto.createHmac("sha256", SECRET_KEY).update(BODY).digest("hex");`

The *Operator* `MUST` check and validate headers and return error in case of incorrect authorization data.

Response codes

All responses `MUST` return appropriate response's codes.

Code	Status
2XX	Successful
4XX	Application Error
5XX	Server Error

Errors

All errors `MUST` follow unified structure. Field `message` should contain reason of the error and it is logged by the *Server* only internally and not passed to the *Game Client* so it can be more descriptive then error messages from the next section.

Example

```
{
  "error": {
    "message": "Token expired",
    "code": "USER_UNAUTHORIZED"
  }
}
```

Error codes

Field `code` is mapped into the following message before passing to the *Game Client*

Code	Message sent to the *Game Client*
USER_UNAUTHORIZED	Couldn't authorize the user
SERVER_UNAUTHORIZED	Couldn't authorize the server
INSUFFICIENT_FUNDS	Not enough money to make withdrawal

Code	Message sent to the *Game Client*
`LOSS_LIMIT`	Loss limit has been exceeded
`UNKNOWN`	Unknown error

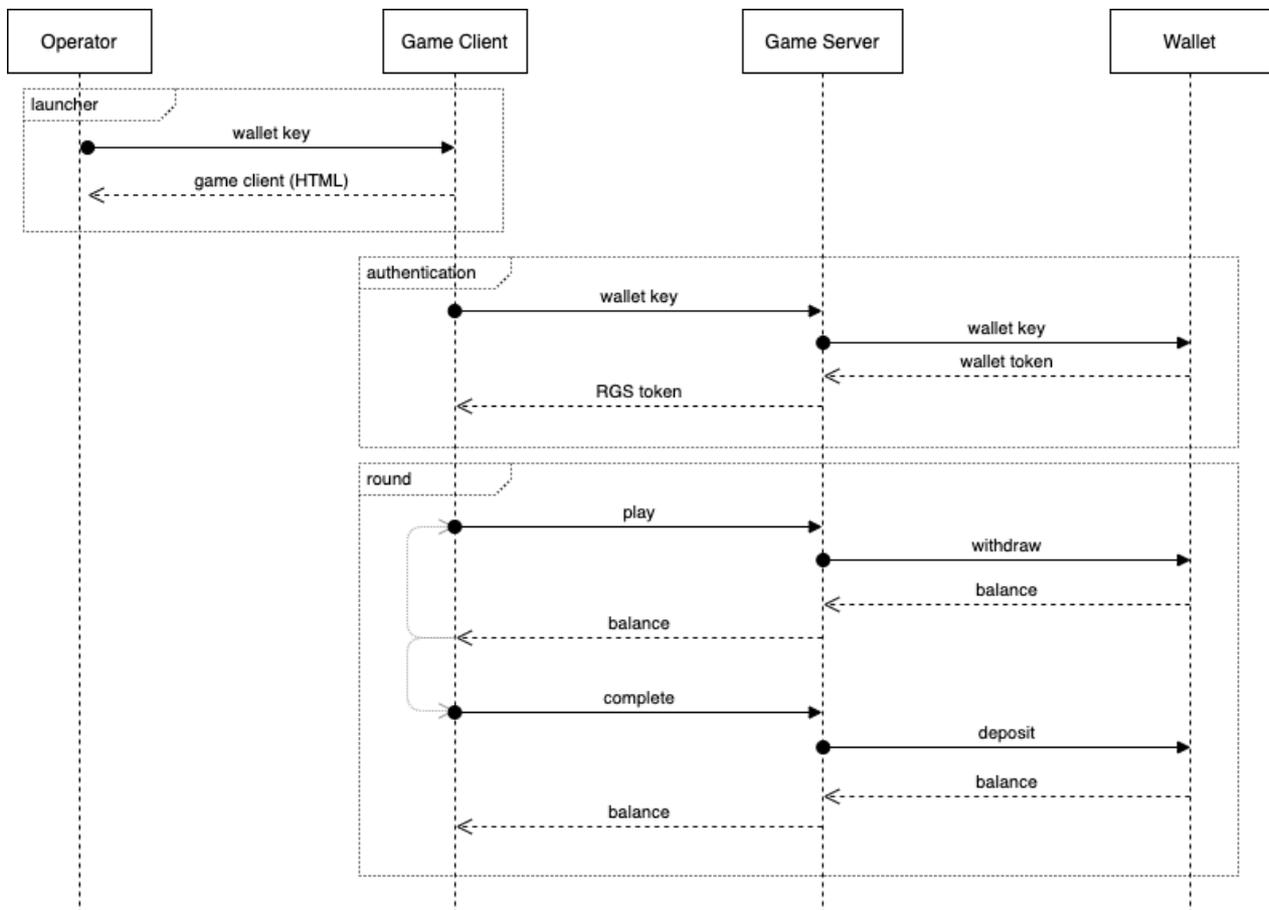## Idempotence

In case of error, network failure or other reasons selected requests `MAY` be repeated therefore selected requests `MUST` be idempotent and return the same outcome.

Automated Wallet Verifier

Automated Wallet Verifier offers running automatic test to check integration correctness. To launch it enter in the browser `<SERVER_URL>/verifier?operator=<operator>&key=<key>` and see status of your integration.

Flow

Launcher

Path: `/launch`
Method: `POST` (recommended) or `GET`
Authorization: `NO`
Idempotent: `NO`

To launch the game the *Operator* should embed `<iframe>` pointing to `<SERVER_URL>/launch` .

Parameters

Name	Type	Required	Example	Description
`key`	`string`	`REQUIRED`	`dnsa89me329jdos`	*Operator*'s session initialisation key which `SHOULD` be active only once after generation and expire after `4 hour`
`provider`	`string`	`REQUIRED`	`myProvider`	*Provider* Id
`operator`	`string`	`REQUIRED`	`myOperator`	*Operator* Id
`game`	`string`	`REQUIRED`	`myGame`	*Game* Id
`language`	`string`	`OPTIONAL`	`en`	*User*'s language code in ISO 639-1 format
`lobbyUrl`	`string`	`OPTIONAL`	`http://lobby.url`	URL to *Operator*'s lobby
`depositUrl`	`string`	`OPTIONAL`	`http://deposit.url`	URL to *Operator*'s in-game deposit webpage

Response

Returns HTML web page containing the game

API

Path: `/authenticate`
Method: `POST`
Authorization: `X-Server-Authorization`
Idempotent: `NO`

Authenticates a *User* and starts a new session.

Parameters

Name	Type	Required	Example	Description
`operator`	`string`	`REQUIRED`	`MyOprator`	*Operator* Id
`key`	`string`	`REQUIRED`	`dnsa89me329jdos`	*Operator*'s session initialisation key which `SHOULD` be active only one time after generation

Response

Name	Type		Example	Description
`nativeId`	`string`	`REQUIRED`	`user123432`	*Operator*'s Player ID
`token`	`string`	`REQUIRED`	`db76b22745bca2cd`	Authorization token which will be sent in `Authorization` token in order to verify user. Preferably it should expire after `7 days`
`balance`	`float`	`REQUIRED`	`100.85`	*User*'s current balance

Name	Type		Example	Description
`currency`	`string`	`REQUIRED`	`eur`	*User's* currency, ***cannot change once set!***
`brand`	`string`	`REQUIRED`	`YellowCasino`	*Operator's* brand
`nickname`	`string`	`OPTIONAL`	`JohnyBravo`	*User's* nickname
`gender`	`string`	`OPTIONAL`	`m`	*User's* gender (`m` or `f`)
`country`	`string`	`OPTIONAL`	`uk`	*User's* country code in ISO 3166-1 Alpha-2 format
`jurisdiction`	`string`	`OPTIONAL`	`mt`	*User's* jurisdiction in case of specific market requirements i.e. `MT` , `UK` , `DK`

Path: `/balance`
Method: `GET`
Authorization: `X-Server-Authorization` and `Authorization`
Idempotent: `YES`

Returns *User's* current balance in *User's* currency.

Parameters

None

Response

Name	Type		Example	Description
`balance`	`float`	`REQUIRED`	`100.85`	*User's* current balance

Path: `/transaction`
Method: `PUT`
Authorization: `X-Server-Authorization` and `Authorization`
Idempotent: `YES`

Withdraws or deposits money from *User's* account.
If transaction with given `transactionId` was already processed then don't modify the account and return current balance.

Parameters

Name	Type		Example	Description
`transactionId`	`string`	`REQUIRED`	`321e4567-e89b-45d3-b594-41234174249`	Transaction ID
`type`	`string`	`REQUIRED`	`withdraw`	`withdraw` reduces *User's* balance. `deposit` increases *User's* balance.

Name	Type		Example	Description
`amount`	`float`	`REQUIRED`	`10.56`	Amount in *User's* currency to withdraw from the account. Withdrawal of 0 (zero) amount `MUST` be supported.
`jackpotAmount`	`float`	`OPTIONAL`	`0.0017`	Part of `amount` which goes to jackpot contribution (in case of withdrawals) or jackpot win (in case of deposits). It is only for information purpose, this value should not affect *User's* balance on top of 'amount'`. Please note contribution can have more then usual 2-decimal places
`game`	`string`	`OPTIONAL`	`superSlot`	Name of the game
`roundId`	`string`	`OPTIONAL`	`123e4567-e89b-12d3-a456-426614174000`	Round ID, one round can have multiple transactions
`category`	`string`	`OPTIONAL`	`promo`	Category of transaction: `normal`, `side`, `tip`, `promo`, `jackpot`
`name`	`string`	`OPTIONAL`	`Royal Match`	Name of the transaction
`promotionType`	`string`	`OPTIONAL`	`free_round`	Type of promotional tool `free_round`, `tournament`, `mystery` (can be more)
`promotionId`	`string`	`OPTIONAL`	`myCampaign123`	Id of promotional campaign

Response

Name	Type		Example	Description
`balance`	`float`	`REQUIRED`	`100.85`	*User's* current balance

Path: `/cancel`
Method: `DELETE`
Authorization: `X-Server-Authorization` and `Authorization`
Idempotent: `YES`

Returns once withdrew or deposited money to original account.
Transaction can be canceled only after withdrawal and before depositing - TO CONFIRM!.
If transaction with given `transactionId` was already canceled then don't modify the account and return current balance.
It `MUST NOT` be possible to create new transactions with `transactionId` which was once cancelled.

Parameters

Name	Type		Example	Description
`transactionId`	`string`	`REQUIRED`	`321e4567-e89b-45d3-b594-41234174249`	Transaction ID

Response

Name	Type		Example	Description
`balance`	`float`	`REQUIRED`	`100.85`	*User*'s current balance