

 MINISTERIO DE DERECHOS SOCIALES, CONSUMO Y AGENDA 2030	REST Service for Player Deposits and Cancellations for Gaming Operators	SECRETARÍA GENERAL DE CONSUMO Y JUEGO DIRECCIÓN GENERAL DE ORDENACIÓN DEL JUEGO
--	---	---

**Specification of the
REST Service for Player Deposits and
Cancellations
for Gaming Operators of the Joint Deposit
Limits System
(SLDC-OPERADORES)**

**Version 1.1.0
September 2026**



1	Objectives	5
1.1	Document Change Log	6
2	Functional Specification	7
2.1	API REST and Operations	7
2.2	Input Parameters.....	8
2.2.1	Deposit.....	8
2.2.2	Cancellation	9
2.3	Service Responses	10
2.3.1	Deposit Response.....	10
2.3.2	Cancellation Response	12
2.3.3	Error Response for Any Operation	12
2.3.4	Debugging in the Test Environment.....	13
3	Security and Operation.....	13
3.1	Operator Authorization	14
3.1.1	Communication Establishment.....	14
3.1.2	Operator Authentication	14
3.1.2.1	How to Obtain the Client Credentials	15
3.1.2.2	How to Call the Identity and Access Management (IAM) System .	16
3.1.2.3	How to Call the SLDC API	19
3.1.2.4	Diagram of the Token-Acquisition Flow	20
3.1.2.5	Validity Period of the Credentials and the Token	20
3.2	Idempotency in Operations	22
3.3	Retry Policy for the Operation	23
4	Business Continuity	25
4.1	Business Continuity.....	25
4.1.1	SLDC Business Continuity Strategies	25
4.1.2	Recovery Time Objective (RTO)	25



4.1.3	Declaration of a Contingency Situation	26
4.1.4	Actions by Operators During the Contingency	26
4.1.4.1	Maintaining Service to the Player	26
4.1.4.2	Retention of Operations Carried Out During the Period the Contingency Situation Is Active	26
4.1.5	Storage of Pending Transactions	26
4.1.6	Prohibition of Duplicates	27
4.1.7	Restoration of Service	27
4.1.8	Post-Contingency Reconciliation	27
4.1.9	Contingency Tests	27
4.2	Retention and Storage of Service Responses	28
5	Description of Environments	29
	Production	29
5.1		29
5.1.1	Evidence Required to Obtain Access to PRO	30
5.2	Tests	30
6	Testing of the service	32
6.1	Objectives	32
7	Appendices	32
7.1	Result and Error Codes	32
7.2	Test data sets in the test environment	33
7.2.1	Test Users per Operator	34
7.2.1.1	Construction of the Player Identifier	34
7.2.1.2	Test Data Associated with the User	36
7.2.1.3	Considerations	37
7.2.2	Shared Test Users for Operators	37
7.2.3	Description of the test cases	37



7.2.3.1	Make a deposit COMPLETA (Complete):	37
7.2.3.2	Make a deposit FALLIDA (Failed):	38
7.2.3.3	Perform cancellation (OK):.....	39
7.2.4	List of generic error cases.....	40
7.2.4.1	Mandatory data not provided	40
7.2.5	Responses in the Testing environment.....	54
7.3	Application Form for Registration or Modification in the Service	57
7.3.1	Request Data	58
7.3.1.1	Types of Request.....	58
7.3.1.2	Environment.....	58
7.3.1.3	Electronic Notification	58



1 Objectives

The purpose of this document is to describe the technical and functional specification of the public interface (REST API) for the communication of player deposit and cancellation operations that the Directorate General for Gambling Regulation (hereinafter, DGOJ) will provide to gaming operators.

This service will allow operators to report deposited amounts or cancel a previously executed deposit operation, returning whether the operation is accepted or rejected according to the player's configuration within the Joint Deposit Limits System (hereinafter, SLDC), in accordance with the Royal Decree 520/2026, of 24 June, amending Royal Decree 1614/2011, of 14 November, which develops Law 13/2011, of 27 May, on the regulation of gambling, with regard to licenses, authorizations, and gambling registries, for the introduction of a joint deposit limits system per player, and Royal Decree 176/2023, of 14 March, which establishes safer gambling environments.

To facilitate the implementation of client software consuming this REST API, the Directorate General for Gambling Regulation will provide operators with a testing environment. The first version of this document describes the different calls that can be made to the REST API, including the required fields and possible responses.



1.1 Document Change Log

Version	Date	Description
1.0.0	JUNE 2026	Original drafting
1.1.0	SEPTEMBER 2026	Changes to response fields. Security and operation. Test users. Business continuity. Credential request.

September 2026



2 Functional Specification

2.1 API REST and Operations

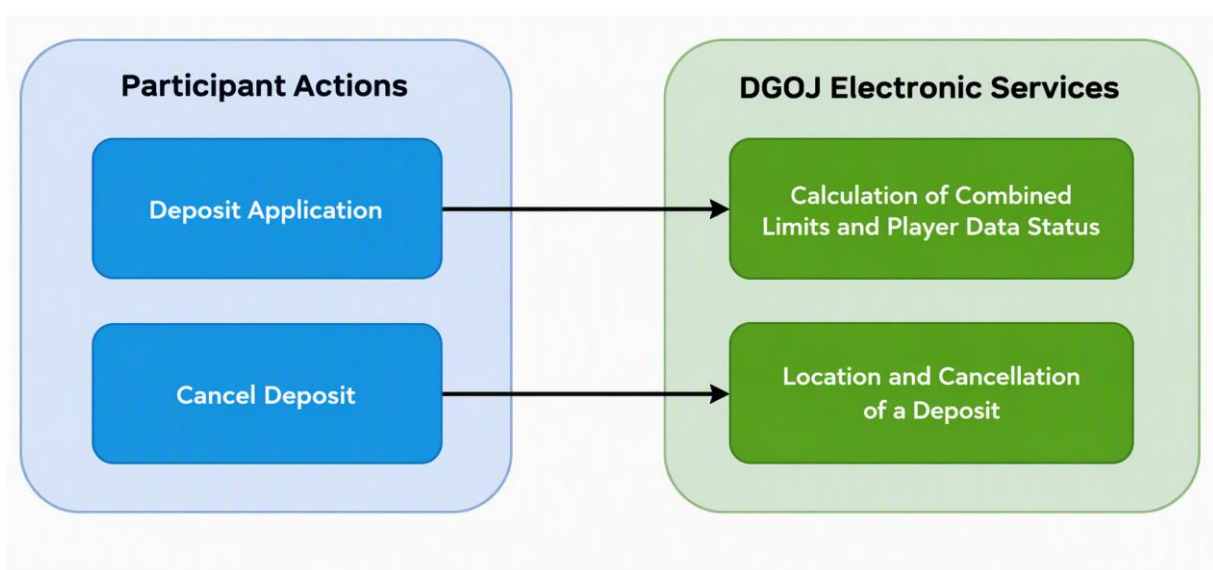
The functional specification of the REST API is defined below:

slcd-operadores		
deposito		
input		depositoRequest
output		depositoResponse
fault		faultResponse
anulacion		
input		anulacionRequest
output		anulacionResponse
fault		faultResponse

The REST API will provide two operations:

1. **“Depósito” (Deposit):** The Operator may use this operation, by identifying itself as a valid Operator within the System and specifying the Player performing the operation along with their identification details, date of birth, deposit amount, and reference code, to report the execution of a deposit.
2. **“Anulación” (Cancellation):** The Operator may use this operation to cancel a deposit previously made by a Player.

An explanatory diagram is shown in the following figure:





2.2 Input Parameters

2.2.1 Deposit

A deposit request submitted by an Operator shall be performed via a POST call to an endpoint, including the following information:

Attribute	Type	Maximum size	Description
uuidOperacionOperador	Uuid (Required)	36 characters	Unique identifier generated by the Operator. It prevents the same operation from being duplicated within the system. Hexadecimal format. One block of 8 characters, three blocks of 4 characters, and a final block of 12 characters. The blocks shall be separated by hyphens.
idOperador	String (Required)	32 characters	Unique identifier of the Operator for the DGOJ.
cantidadDeposito	Integer (Required)	Numeric (Integer)	Amount, in euro cents, to be applied as a deposit for the specified Player.
identificadorJugador	String (Required)	32 characters	Identification document (DNI/NIE) of the Player within the System.
fechaEfectivaOperacion	Datetime (Required)	Format UTC-Zulu: YY/MM/DDTHH:M M:SS.sssZ	Date on which the Operator records the operation in its systems.
fechaNacimiento	Date (Required)	Format: YY/MM/DD	As part of the Player's identification data, the date of birth held by the Operator for the Player.
nombre	String (Required)	64 characters	As part of the Player's identification data, the Player's name as held by the Operator.
primerApellido	String (Required)	64 characters	As part of the Player's identification data, the Player's first surname as recorded by the Operator.
segundoApellido	String (Optional)	64 characters	As part of the Player's identification data, the Player's second surname as recorded by the Operator.



2.2.2 Cancellation

A cancellation request for a deposit submitted by an Operator shall be performed via a POST call to an endpoint, after having received a response to a previous deposit operation with status COMPLETA, and providing the following information:



Attribute	Type	Maximum size	Description
uuidOperacionOperador	Uuid (Required)	36 characters	Unique identifier generated by the Operator. It prevents the same operation from being duplicated within the system. Hexadecimal format. One block of 8 characters, three blocks of 4 characters, and a final block of 12 characters. The blocks shall be separated by hyphens.
idOperador	String (Required)	32 characters	Unique identifier of the Operator for the DGOJ.
identificadorJugador	String (Required)	32 characters	Identification document (DNI/NIE) of the Player within the System.
uuidRegistroOperacion	Uuid (Required)	36 characters	Unique identifier returned by the deposit operation to be cancelled.
fechaEfectivaOperacion	Datetime (Required)	Format UTC-Zulu: YY/MM/DD THH:MM:SS.sssZ	Date on which the Operator records the operation in its systems.
fechaNacimiento	Date (Required)	Format: YY/MM/DD	As part of the Player's identification data, the Player's date of birth as recorded by the Operator.
nombre	String (Required)	64 characters	As part of the Player's identification data, the Player's name as recorded by the Operator.
primerApellido	String (Required)	64 characters	As part of the Player's identification data, the Player's first surname as recorded by the Operator.
segundoApellido	String (Optional)	64 characters	As part of the Player's identification data, the Player's second surname as recorded by the Operator.

2.3 Service Responses

2.3.1 Deposit Response

The deposit endpoint, as a successful response to the request with an HTTP 200 response, shall return the following information:

Attribute	Type	Description
uuidOperacionOperador	Uuid	Unique identifier generated by the Operator for the original operation
uuidRegistroOperacion	Uuid	Unique identifier generated by the System to be used in case it is necessary to subsequently cancel the Operation.
fechaEfectivaOperacion	Datetime	Date on which the System records the requested operation.



estadoOperacion	String (enumeration)	Indicates the result of the requested operation, consisting of: “ COMPLETA ” (COMPLETE): The System accepts and has recorded the requested amount “ FALLIDA ” (FAILED): The requested amount exceeds one or more of the joint limits set by the Player, and therefore the amount is not accepted.
estadoJugador	String (enumeration)	Player Status according to the Player Verification Service. The return codes for this field will correspond to those documented in section 4.1 of the Player Verification Web Service (SVJ) specifications document. If the SLDC has not yet been able to query the status in the SVJ, the status “COD800” will be returned.
descripcionEstadoOperacion	String	For “estadoOperacion” with the value “FALLIDA”, the reason returned will be: “La cantidad solicitada supera alguno de los límites conjuntos fijados de conformidad con el art. 36 bis del RD 1614/2011 no obstante, esta limitación no será efectiva hasta el 26 de marzo de 2027, si desea conocer el estado de sus depósitos conjuntos https://portaljugador.ordenacionjuego.es/limites-conjuntos ”
datosLimitesJugador	Array (optional per environment)	Array containing the player’s status data (only in the Test environment), as described in the “Debugging in the Test Environment” section



2.3.2 Cancellation Response

The cancellation endpoint, as a successful response to the request with an HTTP 200 response, shall not return any payload, except in the Test environment, where it will return a single attribute.

<i>Attribute</i>	<i>Type</i>	<i>Description</i>
uuidOperacionOperador	Uuid	Unique identifier generated by the Operator for the original operation
uuidRegistroOperacion	Uuid	Unique identifier generated by the System for its record.
datosLimitesJugador	Array (optional per environment)	Array containing the player's status data (only in the Test environment), as described in the "Debugging in the Test Environment" section

2.3.3 Error Response for Any Operation

In the event of a failure in the requested operation, the service shall return different HTTP response codes:

<i>HTTP Response Code</i>	<i>Description</i>
4xx	Logical errors in the request.
5xx	Internal connectivity errors.

In both cases, the response payload will contain the following JSON (see section 7.1):

<i>Attribute</i>	<i>Type</i>	<i>Description</i>
codigo	Integer	Identification of the functional error code.
descripcion	String	Description of the detected error.



2.3.4 Debugging in the Test Environment

In the Test environment (see section 5), for both Deposit and Cancellation operations, the player's status data after the requested operation has been applied shall be added for debugging purposes.

This attribute, the `datosLimiteJugador` array, shall contain, for each limit associated with the Player, the following information:

<i>Attribute</i>	<i>Type</i>	<i>Description</i>
<code>idDefinicionLimite</code>	Integer	Unique identifier of the reported limit (daily, weekly, 4 weeks)
<code>cuantiaActual</code>	Integer	Euro cents associated with the player for the specified limit
<code>tipoCambio</code>	String	Internal information on the limit change status (ignore)
<code>fechaFuturaSiguienteCuantia</code>	String Date Time UTC Zulu (optional)	If a future increasing limit is assigned, the date on which it will be applied.
<code>cuantiaFutura</code>	Integer	Euro cents to be applied in the future for the associated limit.
<code>fechaBloqueoLimite</code>	String Date Time UTC Zulu (optional)	Date on which the player will be able to apply new limit increases.
<code>fechaInicio</code>	String DateTime UTC Zulu	Start date of the interval in which the limit begins.
<code>fechaFin</code>	String DateTime UTC Zulu	End date of the interval when the limit will be reset.
<code>acumuladoActual</code>	Integer	Euro cents that the player has already applied in deposits.
<code>etiqueta</code>	String	Readable name of the reported limit (Diario, Semanal...)

3 Security and Operation



3.1 Operator Authorization

3.1.1 Communication Establishment

Communication between the operator's systems and the REST API will be established exclusively via the **HTTPS** protocol, using encryption mechanisms based on **TLS** (with secure versions enabled). This scheme guarantees the confidentiality and integrity of data in transit. Server authentication is performed using the corresponding digital certificates managed by the DGOJ, ensuring that requests and responses travel fully encrypted to prevent unauthorized access or alteration on the network. The operator's client must incorporate the DGOJ's CA certificates into its trust store to allow cryptographic validation of the REST API server.

3.1.2 Operator Authentication

Authentication and authorization between the operator's systems and the REST API will be based on identifying the operator's client system using **client credentials** previously provided through the procedure established on the DGOJ's Electronic Office.

Using these credentials, the client system will make an authentication request to the **Identity and Access Management (IAM)** system, which will verify the client's identity and, if authentication is successful, will issue a **JWT access token**.

This token will allow the client system to prove its identity and authorization to the **API Gateway** of the SLDC service. In each request to the protected services, the operator must present the token obtained, which will be validated by the access infrastructure before access to the requested service is allowed.

This section specifies the procedure a client system (operator) must follow to authenticate against the **Identity and Access Management (IAM)** of the DGOJ, obtain and validate a JWT token, and subsequently present it to the **API Gateway** of the **SLDC service**.

Authentication is structured into the following three blocks:

- How to Obtain the Client Credentials
- How to call the Identity and Access Management (IAM) system in order to obtain a JWT token
- How to call the SLDC API with a valid JWT token.



3.1.2.1 How to Obtain the Client Credentials

To authenticate against the Identity and Access Management (IAM) system and access the services exposed by the API Gateway, each operator must have client credentials associated with its system. In each request to the protected services, the operator must present the token obtained together with the X-Id-Operator header, whose value must match the operator's identity included in the token itself; both elements will be validated by the access infrastructure before access to the requested service is allowed.

To obtain the credentials, the procedure described in section 7.3 will be used.

Once the request has been made and the corresponding checks carried out, the operator will be provided with the Client Secret, a credential which, together with its own operator identifier (used as `client_id`), allows its client system to authenticate against the IAM. These credentials must be kept securely by the operator and used exclusively for authorized communications with the DGOJ's services. This parameter is called the Client Secret.

The Client Secret is a private credential associated with a client system registered in the IAM, used to prove its identity during the authentication process via the OAuth 2.0 Client Credentials Grant flow.

Together with the client's public identifier, (`client_id`), which matches the operator identifier (`idOperator`), the Client Secret allows the IAM to verify that the request to obtain an access token comes from a previously registered and authorized client system.

The Client Secret (`client_secret`) must be considered confidential information. It must not be included in source code, code repositories, application logs, URLs, or other elements that are accessible or susceptible to exposure. It must also be stored and managed using secure mechanisms and must be renewed or revoked in the event of a suspected compromise, or in accordance with the established credential management policy.

The Client Secret does not represent the identity of a user, but rather the identity and authorization of the application or client system that communicates with the DGOJ's services.



3.1.2.2 How to Call the Identity and Access Management (IAM) System

3.1.2.2.1 Endpoint

<i>Field</i>	<i>Value</i>
Method	POST
URL	{BASE_URL}/realms/dgoj/protocol/openid-connect/token
BASE_URL (pre-production)	https://iam-pre.ordenacionjuego.es
BASE_URL (production)	https://iam.ordenacionjuego.es

3.1.2.2.2 Header

<i>Header</i>	<i>Mandatory</i>	<i>Value</i>
Content-Type	Yes	application/x-www-form-urlencoded

3.1.2.2.3 Body parameters (form-urlencoded)

<i>Parameter</i>	<i>Mandatory</i>	<i>Description</i>
grant_type	Yes	Fixed: client_credentials. Indicates an M2M flow with no end user.
client_id	Yes	Identifier of the operator's client, in the format <ID_OPERADOR> (e.g. 1040).
client_secret	Yes	Confidential secret of the client



Parameter	Mandatory	Description
		Sent in the body (client_secret_post).

3.1.2.2.4 Example request

```
BASE_URL="https://iam-pre.ordenacionjuego.es"

CLIENT_ID="1040"

CLIENT_SECRET="<operator_client_secret>"

curl -s -X POST \
  "$BASE_URL/realms/dgoj/protocol/openid-connect/token" \
  -H "Content-Type: application/x-www-form-urlencoded" \
  -d "grant_type=client_credentials" \
  -d "client_id=$CLIENT_ID" \
  -d "client_secret=$CLIENT_SECRET"
```

3.1.2.2.5 Expected response (HTTP 200)

```
{
  "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXLTU1IiwiaWF0Ij0iMTU1MjM0MDAwMCJ9",
  "expires_in": 1800,
  "refresh_expires_in": 0,
  "token_type": "Bearer",
  "not-before-policy": 0,
  "scope": "sldc-operadores"
}
```



<i>Response field</i>	<i>Description</i>
access_token	JWT (RS256) that must be forwarded in the Authorization header of calls to the business API.
expires_in	Seconds for which the token is valid. Once that time has passed, a new one must be requested.
refresh_expires_in	0 in this flow: client_credentials does not issue a refresh_token, as there is no session to refresh.
token_type	Always Bearer; the Authorization header must be prefixed accordingly.
scope	Effective scopes of the token; includes sldc-operadores.



3.1.2.3 How to Call the SLDC API

Once the JWT token (access_token) has been obtained, the operator presents it together with its idOperador in each call to the business endpoints of the SLDC API

3.1.2.3.1 Required headers

Header	Mandatory	Value
Authorization	Yes	Bearer <access_token>
X-Id-Operador	Yes	Must match the idOperador claim of the token (IdOperadorHeader parameter in openapi.yaml.tpl).
Content-Type	Yes	application/json

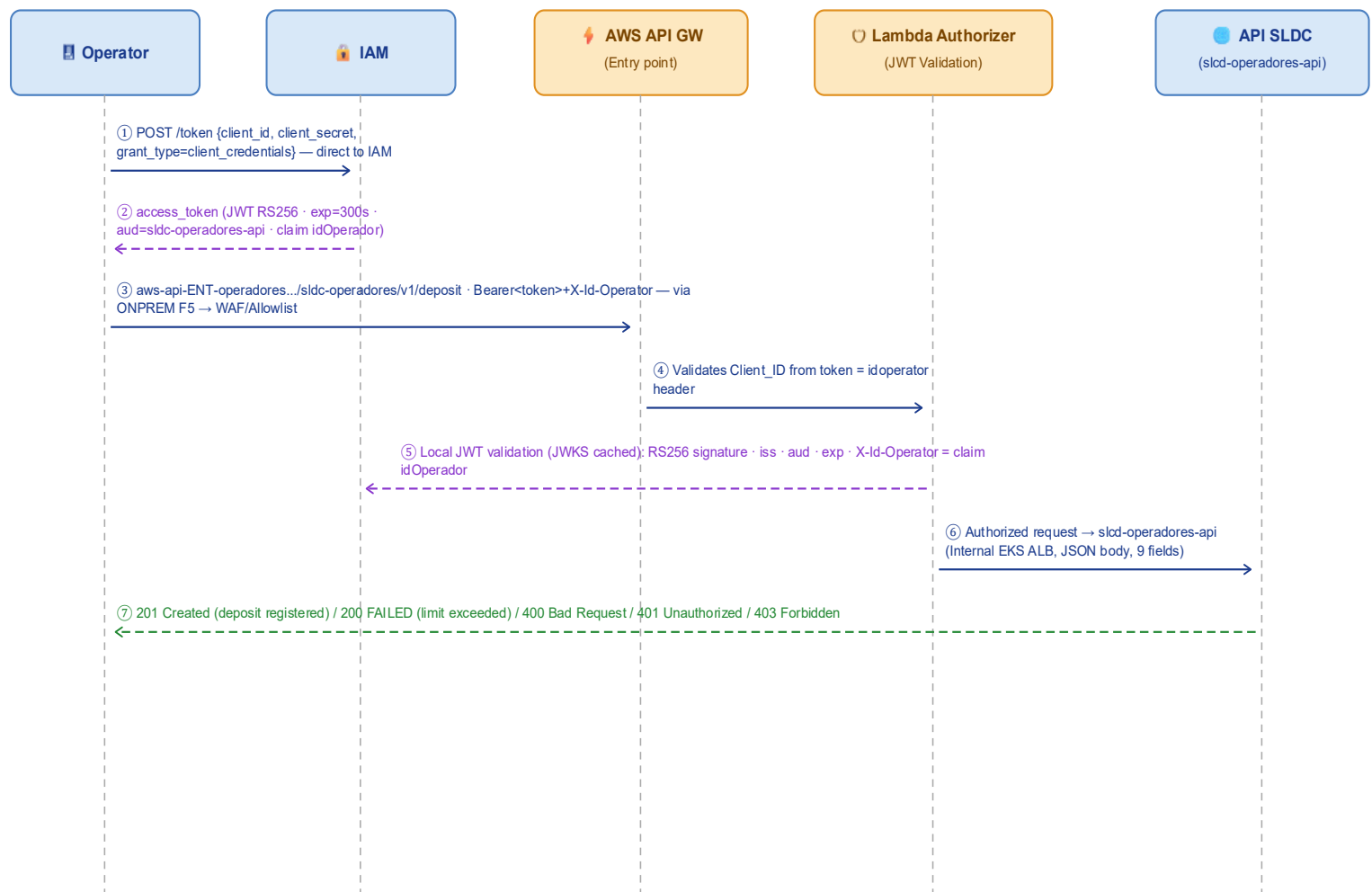
3.1.2.3.2 Example request

```
curl -X POST \  
  "https://aws-api-pre-operadores.ordenacionjuego.es/sldc-operadores/v1/deposito" \  
 \  
  -H "Authorization: Bearer $ACCESS_TOKEN" \  
  -H "X-Id-Operador: 1040" \  
  -H "Content-Type: application/json" \  
  -d '{  
    "uuidOperacionOperador": "f47ac10b-58cc-4372-a567-0e02b2c3d479",  
    "idOperador": "1040",  
    "cantidadDeposito": 100,  
    "identificadorJugador": "12345678A",  
    "fechaEfectivaOperacion": "2025-01-22T00:00:00.000Z",  
    "nombre": "Felipe",  
    "primerApellido": "Herrero",  
    "segundoApellido": "González",  
    "fechaNacimiento": "2000-05-15"  
  }'
```



Note: The body of the requests is detailed in section 2.2 and section 7.2.3

3.1.2.4 Diagram of the Token-Acquisition Flow



3.1.2.5 Validity Period of the Credentials and the Token

Client credentials (client_secret) will have a **validity period of one year** from their date of issue.

Using these credentials, the client system may request from the authentication system a **access token (JWT)**. Each access token will have a **validity period of 30 minutes** from the time it is issued.



Once the access token has expired, the client system must request a new token through a new authentication using the current client credentials.

Renewing the access token does not imply renewal of the client credentials, which will retain their one-year validity period unless they are revoked or replaced in accordance with the established procedure.



3.2 Idempotency in Operations

In order to ensure that the same operation is not processed more than once, the service will use, as indicated in section “2.2 Input parameters”, the `uuidOperacionOperador` field as the unique identifier for each operation.

The `uuidOperacionOperador` will be generated by the operator’s client system and must be unique for each new operation. This identifier will allow the SLDC to distinguish between a new operation and a retry of an operation that has already been sent previously.

If the operator does not receive a response to a request, for example due to a timeout or a communications interruption, and needs to retry, it must use the same `uuidOperacionOperador` used in the original request. This way, the SLDC can check whether the operation has already been processed and avoid duplicate execution.

Conversely, when it is a new operation, the operator must generate a new `uuidOperacionOperador`.

The identifier must comply with the format established in section “2.2 Input parameters”:

The idempotency mechanism therefore makes it possible to safely manage retries without the same operation producing duplicate effects in the system.

Rules of use

<i>Situation</i>	<i>uuidOperacionOperador</i>
New deposit or cancellation operation	Generate a new
Retry of the same deposit or cancellation operation	Keep the same
New operation following a previous deposit or cancellation operation	Generate a new
Timeout with unknown outcome	Retry with the same
Functional error preventing the operation from being processed	Do not generate a new one for the same attempt
Same key with different request content	The SLDC will reject the request



3.3 Retry Policy for the Operation

In the event of a transient error during communication with the DGOJ's services, the client system must apply a retry strategy based on **exponential backoff with jitter**, in order to avoid simultaneous retries and reduce the load on the affected services.

The strategy will be limited to a **maximum of 3 retry cycles**. Each cycle will consist of an **initial attempt and four additional retries**, applying the following wait intervals:

Attempt	Wait interval
---------	---------------

Initial attempt	0 seconds
-----------------	-----------

Retry 1	1 second + jitter
---------	-------------------

Retry 2	2 seconds + jitter
---------	--------------------

Retry 3	4 seconds + jitter
---------	--------------------

Retry 4	8 seconds + jitter
---------	--------------------

The **jitter** will consist of a random variation added to the established wait interval, in order to prevent different client systems from performing their retries in a synchronized manner.

If, after completing a cycle, no satisfactory response is obtained, the system will perform a **wait of 5 minutes** before starting the next cycle.

The sequence will therefore be as follows:

Cycle 1

Initial attempt → 1 s + jitter → 2 s + jitter → 4 s + jitter → 8 s + jitter → **wait of 5 minutes**

Cycle 2

Initial attempt → 1 s + jitter → 2 s + jitter → 4 s + jitter → 8 s + jitter → **wait of 5 minutes**

Cycle 3

Initial attempt → 1 s + jitter → 2 s + jitter → 4 s + jitter → 8 s + jitter

After completing the third cycle without obtaining a satisfactory response, no further automatic retries will be performed and the operation will be considered failed, and the error handling defined for the service must be applied.

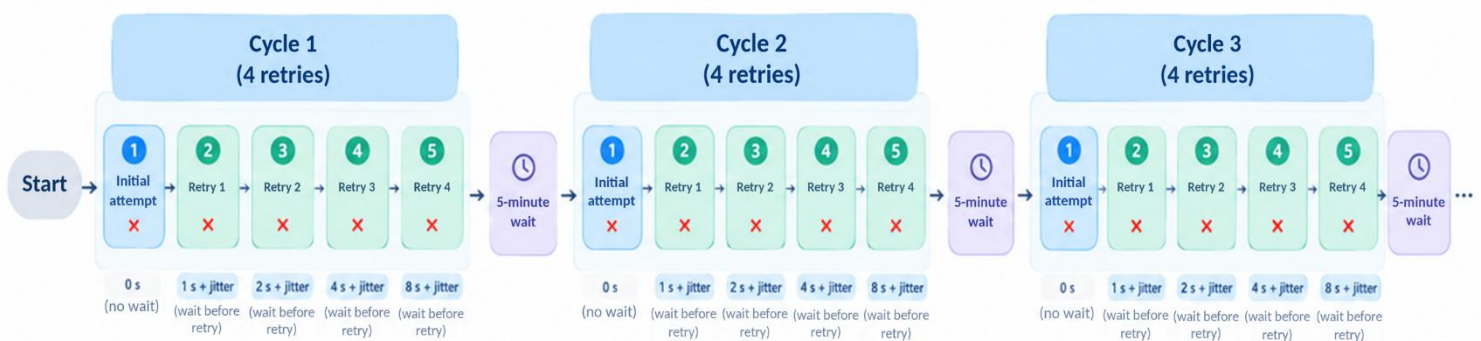


Retries must only be carried out in situations that are potentially recoverable, such as temporary communication errors, temporary service unavailability, or certain transient HTTP errors. Functional or request-validation errors must not trigger automatic retries.

For operations that may produce effects on the system, all retries corresponding to the same operation must retain the same `uuidOperacionOperador` value, thereby ensuring the operation's idempotency and preventing duplicate processing.

Retry Strategy

Exponential backoff with jitter • 4 retries per cycle • 5-minute wait between cycles



Exponential backoff with jitter

- The wait time between retries increases exponentially: 0 s, 1 s, 2 s, 4 s, and 8 s.
- Jitter (random variation) is applied to prevent multiple clients from retrying at the same time.
- The values shown are indicative and can be adjusted according to the integration policy.



Retry cycle

- 4 retries are performed per cycle.
- After the 4 retries are completed, the process waits 5 minutes before starting a new cycle.
- The process repeats indefinitely until the operation succeeds or a defined retry limit is reached.



Flow summary

- 1 Initial attempt (no wait)
 - 2 Retry 1 (1 s + jitter)
 - 3 Retry 2 (2 s + jitter)
 - 4 Retry 3 (4 s + jitter)
 - 5 Retry 4 (8 s + jitter)
- 5-minute wait
- Repeat cycle

NOTE: These values have been established as reference parameters for the initial configuration of the system. These values may be adjusted during performance, load, and resilience testing, based on the results obtained and the specific characteristics of each operation.



4 Business Continuity

4.1 Business Continuity

The Joint Deposit Limits System (SLDC) has been designed to guarantee high availability and ensure the continuous provision of the service to participating operators. However, exceptional technical incidents may occur that temporarily affect the normal provision of the service.

This section describes the SLDC's business continuity strategies and the contingency measures operators must apply when a service interruption exceeds the Recovery Time Objective (RTO) defined for the system.

4.1.1 SLDC Business Continuity Strategies

The SLDC platform has technical and organizational mechanisms designed to minimize the impact of incidents that could compromise the availability of the service, including:

- Infrastructure deployed in redundant environments.
- Replication of critical components and data storage.
- Continuous monitoring of services and their performance.
- Disaster recovery procedures (Disaster Recovery).
- Service restoration capabilities at an alternative location.
- 24x7 incident monitoring and management.
- Technical and operational escalation procedures.

These measures aim to restore normal operations within the committed timeframes for the service.

4.1.2 Recovery Time Objective (RTO)

The RTO represents the maximum expected time to restore SLDC operations following a serious service interruption. The declaration of a serious interruption will be notified by the DGOJ.

While the incident continues within the established RTO, operators must suspend the usual retry and error-handling mechanisms provided for in the technical integration, as well as deposit queries, while awaiting notification of service recovery or activation of the contingency mechanism.

Exceeding the RTO will be notified by the DGOJ and will formally activate the contingency procedure described in the following sections.



4.1.3 Declaration of a Contingency Situation

When an incident affects the SLDC services and it is estimated that the recovery time will exceed the established RTO, the body responsible for the system will:

1. Declare the contingency status.
2. Communicate this circumstance to all affected operators through the established channels.
3. Report, as far as possible, on the estimated scope of the incident and the progress of the recovery tasks.
4. Subsequently notify the end of the contingency situation and the return to normal operation.

Communications may be made by email, integration portal, enabled notification systems, or any other officially communicated channel.

4.1.4 Actions by Operators During the Contingency

While the contingency situation remains active, operators must adopt the following measures:

4.1.4.1 Maintaining Service to the Player

Operators may continue providing their gambling services, accepting and recording all deposit operations.

4.1.4.2 Retention of Operations Carried Out During the Period the Contingency Situation Is Active

Any operation relating to:

- Deposit operations.
- Deposit cancellation operations.
- Any other transaction regulated through integration with the SLDC.

must be fully recorded by the operator, ensuring its traceability and subsequent reconciliation.

4.1.5 Storage of Pending Transactions

Requests that cannot be processed due to SLDC unavailability must be kept in a secure queue of pending operations for subsequent submission once the service has been restored.

The records must retain:

- Unique identifier of the operation.
- Date and time of generation.
- Data necessary for its reprocessing.



- Result of the attempts made.
- Transaction status.

NOTE: This information will be reviewed after the testing phase

4.1.6 Prohibition of Duplicates

Operators must ensure idempotency mechanisms that prevent multiple processing of the same operation during resubmission and recovery processes.

4.1.7 Restoration of Service

Once SLDC recovery has been declared, operators must:

1. Reactivate automatic integrations with the system.
2. Process, in an orderly manner, the pending operations stored during the contingency period via the specific REST API for reconciling operations carried out during the contingency.
3. Apply the synchronization mechanisms defined by the SLDC.
4. Resolve any discrepancies detected during the reconciliation process through the operator support contact.
5. Verify the correct processing of all pending transactions.

During this phase, the SLDC may apply load-control or request-prioritization measures in order to avoid degradation resulting from the simultaneous recovery of multiple operators.

4.1.8 Post-Contingency Reconciliation

Once the contingency situation has ended, operators must retain the records associated with the incident for the retention period established by current regulations and section 4.2 of these specifications, and must cooperate, when required, in the validation and reconciliation processes carried out by the DGOJ.

The recorded information must make it possible to fully reconstruct the sequence of operations carried out during the period of unavailability, including the resubmission, synchronization, and regularization actions performed after the service has been restored.

4.1.9 Contingency Tests

The DGOJ may periodically call business continuity exercises and contingency tests in order to verify:

- The correct activation of the defined procedures.



- The capacity to store and reprocess pending operations.
- The operation of the synchronization mechanisms.
- The effectiveness of operators' recovery procedures.

Operators must take part in these tests in accordance with the instructions notified.

4.2 Retention and Storage of Service Responses

The operator's client system must retain the responses received from the DGOJ's services, together with the information necessary to identify and contextualize each operation, for a **period of 4 years**.

During the first **12 months**, the information must be kept in a storage system that allows it to be queried and retrieved in an accessible manner, without the need to resort to specific retrieval processes from long-term storage.

Once this period has elapsed, the information must be retained for an additional period of **3 years in long-term storage**, ensuring its integrity and the possibility of retrieval throughout this period whenever necessary.

The retained information must, at a minimum, make it possible to identify and link each response to the corresponding operation. To this end, the information necessary to guarantee the traceability of the operation must be retained, including, where applicable, the identifier **uuidOperacionOperador**, the field **uuidRegistroOperacion**, the date and time of the operation, the request made, the response received, and the information associated with the communication with the service.

The format used for retention must be interoperable and based on widely used standards or formats, so that the information can be consulted, processed, exported, or migrated to other systems without depending on a specific proprietary technology or product.

The specific storage format may be defined by the operator, provided it meets the interoperability, integrity, traceability, readability, and recovery requirements established in this section.

Client credentials, the Client Secret, access tokens, or any other authentication information that allows access to the DGOJ's services must not be stored as part of these records.



5 Description of Environments

The services shall be invoked via URLs within a technological stack that is currently being finalized and validated.

5.1 Production

A maximum of two versions of the service may coexist to allow operators to progressively deploy a version of the service in their production systems. The DGOJ will agree with the operators on the version change schedule, defining which service endpoint and REST version are supported at any given time, as well as the decommissioning timeline for the previous version.

From version 1.0 of the service onwards, the access URLs shall include the corresponding version identifier (v1), coexisting with the URLs of subsequent versions that may be published, during the period established by the DGOJ, with the aim of facilitating the implementation and adaptation of operators.

The version numbering of the document does not correspond to that of the software and the REST API.

For each operator, the environment shall not be activated until a specific license has been granted, and registration for the service may not be requested until that time via the form detailed in section 7.3.

The endpoints for the production environment are:

<https://aws-api-operadores.ordenacionjuego.es/slhc-operadores/v1/deposito>

<https://aws-api-operadores.ordenacionjuego.es/slhc-operadores/v1/anulacion>

Access to the production REST API shall only be enabled for the IP addresses communicated by operators through the procedure indicated in section 4.4.1.3.1 of the specifications document of the [Player Verification Web Service \(SVJ\)](#). The SLHC will have the same source IPs as the SVJ for each operator. These IP addresses shall correspond to the servers or systems that establish communication with the DGOJ and must therefore be individual addresses, with a maximum of four allowed unless a duly justified request is submitted. In the event that the DGOJ decides to suspend the service provided to an operator, the declared IP addresses will be used to block traffic.



It is therefore the operator's responsibility to use unique IP addresses that are not shared with other operators when accessing DGOJ services.

In addition, to obtain access credentials, they must be requested in accordance with section 7.3. To receive them, evidence must be provided that the API of the test environment described in section 5.2 has been tested. The following section explains the necessary requirements

5.1.1 Evidence Required to Obtain Access to PRO

As a prerequisite for access to the **PRO** environment and obtaining the corresponding credentials, the operator must, in the **PRE** environment, carry out the necessary tests on the operator's test user.

The operator must send the DGOJ the responses generated by the service for the following scenarios:

- **COMPLETA deposit operation.**
- **FALLIDA deposit operation.**
- **Cancelled operation.**

The responses must correspond to operations carried out using the **test DNI assigned to the operator** described in section 7.2.1, and must make it possible to verify the correct functioning of the integration, as well as the client system's appropriate handling of the different operation outcomes described in section 2.

These tests must be attached to the procedure described in section 7.3 for requesting access credentials for the **PRO** environment. To complete the request, it will be necessary to provide the responses obtained for the operations indicated above.

Once these tests have been satisfactorily verified by the DGOJ, access to the **PRO** environment may proceed, together with the provision of the credentials necessary to carry out operations in that environment.

5.2 Tests

There is a single access point to the latest version of the service. Version changes will be announced to operators well in advance, and the previous version will be discontinued.

Endpoint

<https://aws-api-pre-operadores.ordenacionjuego.es/sldc-operadores/v1/deposito>

<https://aws-api-pre-operadores.ordenacionjuego.es/sldc-operadores/v1/anulacion>



Where v is a digit indicating the version number of the REST API.

Gaming operators wishing to perform tests with the REST API must request access to the test environment through the form detailed in section 7.3.

The applicant company must be registered at least in the Special Section of Participants of the General Register of Gambling Licenses, prior to the granting of a specific license.

Access to the service does not include IP address restrictions; authentication using credentials in accordance with the procedure mentioned above is sufficient.

To resolve technical incidents or queries, an email must be sent to dgoj.soproteoperadores@ordenacionjuego.gob.es, indicating "[SLDC] Pruebas entorno TEST" in the subject line.



6 Testing of the service

6.1 Objectives

In future versions of the SLDC, operators using the REST API may be provided with the option to check the service status through a dedicated TEST service independent from the SLDC REST API.

7 Appendices

7.1 Result and Error Codes

The following table shows the codes corresponding to the result types of the different operations.

Result Codes	
<i>Deposits</i>	
10000	Affiliation details not provided. The operation is not permitted without this information.
10001	Request denied due to missing data in the operation.
20000	Operation already recorded previously.
90000	Service temporarily unavailable.
<i>Cancellations</i>	
10000	Affiliation details not provided. The operation is not permitted without this information.
10001	Request denied due to missing data in the operation.
20001	Cancellation data not provided or incorrect.
20002	Invalid operation record for a cancellation. A non-existent operation record or one already used previously for a cancellation has been provided.
90000	Service temporarily unavailable. Error in third-party service

The following table shows the codes that will be used for technical error types:



HTTP Code	Status	Action
400 Bad Request	Invalid data	DO NOT retry. Check payload
403 Forbidden	TTL: 1800s / Invalid JWT	Renew token and retry
403 Forbidden	X-Id-Operator does not match the claim	Check configuration. DO NOT retry until fixed
403 Forbidden	Unauthorized IP / JWT without permission	DO NOT retry. Contact DGOJ
408 Request Timeout	Server not responding	Retry with backoff
409 Conflict	Duplicate operation	DO NOT retry (idempotent)
429 Too Many Req.	Rate limit	Wait and retry (Retry-After / 2s)
500 Server Error	Transient error	Retry with backoff
502 Bad Gateway	Backend down/restarting	Retry with backoff
503 Unavailable	Service down	Retry with backoff (Retry-After / 1s)
504 Gateway Timeout	Backend not responding	Retry with backoff

Two important points regarding possible errors:

- This API does not return 401. An expired token (TTL: 1800 s) or an incorrect X-Id-Operator header will generate a 403. On receiving a 403, analyze the JWT to determine how to proceed; renew the token if applicable before escalating
- For 429 and 503 errors: respect the Retry-After header if it is included in the response. If not, apply backoff starting from 2 s.

7.2 Test data sets in the test environment

Endpoint

<https://aws-api-pre-operadores.ordenacionjuego.es/sldc-operadores/v1/deposito>

<https://aws-api-pre-operadores.ordenacionjuego.es/sldc-operadores/v1/anulacion>



Where v is a digit indicating the version number of the REST API.

Testing of the service is based on the basic operations using test users as described in section 7.2.1.

The purpose of the tests is therefore to verify the connectivity of the operator's system with the service, its authentication mechanisms, secure message traffic, the use of the messages exchanged, and the basic operations.

Likewise, stress tests involving large volumes of data must not be carried out against the test environment without prior notice to the DGOJ.

To resolve technical incidents or queries, an email must be sent to dgoj.sopORTEoperadores@ordenacionjuego.gob.es, indicating "[SLDC] Pruebas entorno TEST" in the subject line.

7.2.1 Test Users per Operator

This section defines the information that must be sent in the test requests to the POST /deposito and POST /anulacion services in order to uniquely identify each operator by means of a specific test user.

The aim is for all operators to be able to independently generate their own test data and use it consistently during the system's functional and technical validations.

Each operator will have a single test user associated with its operator identifier (idOperador).

The relationship between operator and test user will be deterministic, so that it can be regenerated at any time without the need to exchange additional information.

7.2.1.1 Construction of the Player Identifier

In requests to the POST /deposito service, the identificadorJugador field must be constructed from the value sent in the idOperador field.

The composition of the identificadorJugador will be:

9999 + idOperador + NIF letter



Where:

- 9999 is a fixed prefix.
- idOperador is the operator identifier sent in the request.
- NIF letter is the letter calculated using the official Spanish NIF algorithm.

Conceptual example

If the request contains:

idOperador = 1083

The base number used for the calculation will be:

99991083

The corresponding NIF letter will be calculated from that number.

The final value of identificadorJugador will be:

99991083X

The specific letter will depend on the result of the calculation algorithm.

NIF letter calculation algorithm

The letter is obtained by applying the standard algorithm used to generate the Spanish NIF.

Process:

1. Concatenate the fixed prefix "9999" with the value of idOperador.
2. Interpret the result as an integer.

September 2026



3. Calculate the remainder of division by 23.
4. Obtain the corresponding letter from the following sequence:

TRWAGMYFPDXBNJZSQVHLCKE
5. Append the obtained letter to the end of the calculated number.

Correspondence examples

idOperador: 1083 identificadorJugador: 99991083 + calculated letter

idOperador: 1029 identificadorJugador: 99991029 + calculated letter

idOperador: 1010 identificadorJugador: 99991010 + calculated letter

idOperador: 1074 identificadorJugador: 99991074 + calculated letter

idOperador: 1098 identificadorJugador: 99991098 + calculated letter

7.2.1.2 Test Data Associated with the User

For all test users generated in accordance with this specification, the system will return the following pre-configured results:

Identity validation result

Code:

COD003

Description:

The user's identity is correct.

Date of birth

Any date in dd/MM/yyyy format may be used for test requests:

11/11/1999



7.2.1.3 Considerations

- There must be a one-to-one relationship between idOperador and identificadorJugador.
- The identificadorJugador must always be calculated from the idOperador using the rules above.
- The generation must be completely deterministic to allow the data to be regenerated at any time.
- All users generated using this specification will return the COD003 identity validation.
- These identifiers are intended for use exclusively in technical, functional, and integration testing.
- Real people's data must not be used during test execution.

7.2.2 Shared Test Users for Operators

For testing purposes, **shared test users for operators will be available.**

To request these users, an email must be sent to dgoj.soproteoperadores@ordenacionjuego.gob.es, indicating "[SLDC] Usuario pruebas compartido" in the subject line.

7.2.3 Description of the test cases

NOTE: To correctly perform these POSTs, the headers for operator authentication indicated in "section 3.1.2", and specifically in "section 3.1.2.3 How to make the call to the SLDC API", must be added

7.2.3.1 Make a deposit COMPLETA (Complete):

POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/deposito>



```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "cantidadDeposito": 300,  
  "identificadorJugador": "X0000000T",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 201 Created

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "uuidRegistroOperacion": "79647742-f937-4dd2-d165-36405ddc715d",  
  "fechaEfectivaOperacion": "2026-06-22T09:48:55.352Z",  
  "estadoOperacion": "COMPLETA",  
  "estadoJugador": "COD800"  
}
```

7.2.3.2 Make a deposit FALLIDA (Failed):

POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/deposito>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d265-36405ddc715d",  
  "idOperador": "1040",  
  "cantidadDeposito": 30000,  
  "identificadorJugador": "X0000000T",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
}
```



```
"fechaNacimiento": "1900-01-01"  
}
```

Response:

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d265-36405ddc715d",  
  "uuidRegistroOperacion": "79647742-f937-4dd2-d165-36405ddc715d",  
  "fechaEfectivaOperacion": "2026-06-22T09:48:55.352Z",  
  "estadoOperacion": "FALLIDA",  
  "estadoJugador": "COD800",  
  "descripcionEstadoOperacion": "La cantidad solicitada supera alguno de los límites conjuntos fijados de conformidad con el art. 36 bis del RD 1614/2011 no obstante, esta limitación no será efectiva hasta el 26 de marzo de 2027, si desea conocer el estado de sus depósitos conjuntos https://portaljugador.ordenacionjuego.es/limites-conjuntos"  
}
```

7.2.3.3 Perform cancellation (OK):

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "identificadorJugador": "X0000000T",  
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 200 OK



```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d265-36405ddc715d",  
  "uuidRegistroOperacion": "796556-f937-4dd2-d165-36405ddc715d"  
}
```

7.2.4 List of generic error cases

7.2.4.1 Mandatory data not provided

If any of the data fields defined as mandatory in the service are missing, the errors indicated in section 5.1 will be returned.

1.1.1.1.1 Deposit with "Uuid" field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/deposito>

```
{  
  "idOperador": "1040",  
  "cantidadDeposito": 30000,  
  "identificadorJugador": "X0000000T",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "codigo": 10001,  
    "descripcion": "Deposit data not provided or incorrect"  
  }  
}
```



1.1.1.1.2 Deposit with “idOperador” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/deposito>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc735d",
  "cantidadDeposito": 300,
  "identificadorJugador": "X0000000T",
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",
  "nombre": "PRUEBAS",
  "primerApellido": "ESPANOL",
  "segundoApellido": "NIEPRUEBAS",
  "fechaNacimiento": "1900-01-01"
}
```

Response:

Code HTTP: 400 Bad Request

```
{
  "resultadoOperacion": {
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
    "codigo": 10001,
    "descripcion": "Deposit data not provided or incorrect"
  }
}
```

1.1.1.1.3 Deposit with “cantidadDeposito” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/deposito>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "idOperador": "1040",
  "identificadorJugador": "X0000000T",
}
```

September 2026



```
"fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
"nombre": "PRUEBAS",  
"primerApellido": "ESPANOL",  
"segundoApellido": "NIEPRUEBAS",  
"fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
    "codigo": 10001,  
    "descripcion": "Deposit data not provided or incorrect"  
  }  
}
```

1.1.1.1.4 Deposit with “identificadorJugador” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/deposito>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "cantidadDeposito": 300,  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

September 2026



```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
    "codigo": 10000,  
    "descripcion": "Personal details not provided"  
  }  
}
```

1.1.1.1.5 Deposit with “fechaEfectivaOperacion” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/deposito>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "cantidadDeposito": 300,  
  "identificadorJugador": "X0000000T",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
    "codigo": 10001,  
    "descripcion": "Deposit data not provided or incorrect"  
  }  
}
```

1.1.1.1.6 Deposit with “fechaNacimiento” field not provided



POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/deposito>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "idOperador": "1040",
  "cantidadDeposito": 300,
  "identificadorJugador": "X0000000T",
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",
  "nombre": "PRUEBAS",
  "primerApellido": "ESPANOL",
  "segundoApellido": "NIEPRUEBAS",
}
```

Response:

Code HTTP: 400 Bad Request

```
{
  "resultadoOperacion": {
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
    "codigo": 10000,
    "descripcion": "Personal details not provided"
  }
}
```

1.1.1.1.7 Deposit with “nombre” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/deposito>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "idOperador": "1040",
  "cantidadDeposito": 300,
  "identificadorJugador": "X0000000T",
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",
}
```

September 2026



```
"primerApellido": "ESPAÑOL",  
"segundoApellido": "NIEPRUEBAS",  
"fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
    "codigo": 10000,  
    "descripcion": "Personal details not provided"  
  }  
}
```

1.1.1.1.8 Deposit with “primerApellido” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/deposito>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "cantidadDeposito": 300,  
  "identificadorJugador": "X0000000T",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {
```

September 2026



```
"uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
"codigo": 10000,  
"descripcion": "Personal details not provided"  
}  
}
```

1.1.1.1.9 Deposit with “segundoApellido” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/deposito>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "cantidadDeposito": 300,  
  "identificadorJugador": "X0000000T",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Since it is not a mandatory field, the operation is accepted.

Code HTTP: 201 Created

```
{  
  "uuidRegistroOperacion": "79647742-f937-4dd2-d165-36405ddc715d",  
  "fechaEfectivaOperacion": "2026-06-22T09:48:55.352Z",  
  "estadoOperacion": "COMPLETA",  
  "estadoJugador": "COD800"  
}
```

1.1.1.1.10 Deposit with data from an already processed operation

POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/deposito>

September 2026



Response:

Code HTTP: 400 Bad Request

```
{
  "resultadoOperacion": {
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
    "codigo": 20000,
    "descripcion": "Operation already previously recorded"
  }
}
```

1.1.1.1.11 Cancellations with “Uuid” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>

```
{
  "idOperador": "1040",
  "identificadorJugador": "X0000000T",
  "uuidRegistroOperacion": "79647742-f937-4dd2-d165-36405ddc715d",
  "nombre": "PRUEBAS",
  "primerApellido": "ESPANOL",
  "segundoApellido": "NIEPRUEBAS",
  "fechaNacimiento": "1900-01-01"
}
```

Response:

Code HTTP: 400 Bad Request

```
{
  "resultadoOperacion": {
    "codigo": 20001,
    "descripcion": "Cancellation data not provided or incorrect"
  }
}
```



1.1.1.1.12 Cancellations with “idOperador” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "identificadorJugador": "X0000000T",
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",
  "nombre": "PRUEBAS",
  "primerApellido": "ESPANOL",
  "segundoApellido": "NIEPRUEBAS",
  "fechaNacimiento": "1900-01-01"
}
```

Response:

Code HTTP: 400 Bad Request

```
{
  "resultadoOperacion": {
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
    "codigo": 20001,
    "descripcion": "Cancellation data not provided or incorrect"
  }
}
```

1.1.1.1.13 Cancellations with “identificadorJugador” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "idOperador": "1040",
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",
  "nombre": "PRUEBAS",
}
```

September 2026



```
"primerApellido": "ESPAÑOL",  
"segundoApellido": "NIEPRUEBAS",  
"fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
    "codigo": 10000,  
    "descripcion": "Personal details not provided"  
  }  
}
```

1.1.1.1.14 Cancellations with "uuidRegistroOperacion" field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "identificadorJugador": "X0000000T",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPAÑOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
```

September 2026



```
"codigo": 20001,  
"descripcion": "Cancellation data not provided or incorrect"  
}  
}
```

1.1.1.1.15 Cancellations with incorrectly specified "uuidRegistroOperacion" field

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "identificadorJugador": "X0000000T",  
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b95",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
    "codigo": 20002,  
    "descripcion": "Invalid operation record or inconsistent with the identifier"  
  }  
}
```

1.1.1.1.16 Cancellations with "fechaEfectivaOperacion" field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>



```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "identificadorJugador": "X0000000T",  
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

The operation is accepted without the field

Response:

Code HTTP: 200 OK

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d265-36405ddc715d",  
  "uuidRegistroOperacion": "796556-f937-4dd2-d165-36405ddc715d",  
}
```

1.1.1.1.17 Cancellations with “fechaNacimiento” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/anulacion>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "identificadorJugador": "X0000000T",  
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
}
```



Response:

Code HTTP: 400 Bad Request

```
{
  "resultadoOperacion": {
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
    "codigo": 10000,
    "descripcion": "Personal details not provided"
  }
}
```

1.1.1.1.18 Cancellations with “nombre” field not provided

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/anulacion>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "idOperador": "1040",
  "identificadorJugador": "X0000000T",
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",
  "primerApellido": "ESPAÑOL",
  "segundoApellido": "NIEPRUEBAS",
  "fechaNacimiento": "1900-01-01"
}
```

Response:

Code HTTP: 400 Bad Request

```
{
  "resultadoOperacion": {
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
    "codigo": 10000,
    "descripcion": "Personal details not provided"
  }
}
```



1.1.1.1.19 Cancellations with “primerApellido” field not provided

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "identificadorJugador": "X0000000T",  
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 400 Bad Request

```
{  
  "resultadoOperacion": {  
    "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
    "codigo": 10000,  
    "descripcion": "Personal details not provided"  
  }  
}
```

1.1.1.1.20 Cancellations with “segundoApellido” field not provided

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "identificadorJugador": "X0000000T",  
  "uuidRegistroOperacion": "019eda7d-b831-75d7-a2c9-7e38cfb7b957",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "fechaNacimiento": "1900-01-01"  
}
```



Response:

As it is not a mandatory field, the operation is accepted

Code HTTP: 200 OK

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d265-36405ddc715d",  
  "uuidRegistroOperacion": "796556-f937-4dd2-d165-36405ddc715d",  
}
```

7.2.5 Responses in the Testing environment

In the Testing environment, a more complete response is provided to assist with test management

1.1.1.1.21 Deposit Response

POST <https://aws-api-pre.ordenacionjuego.es/sldc-operadores/v1/deposito>

```
{  
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",  
  "idOperador": "1040",  
  "cantidadDeposito": 500,  
  "identificadorJugador": "X0000000T",  
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",  
  "nombre": "PRUEBAS",  
  "primerApellido": "ESPANOL",  
  "segundoApellido": "NIEPRUEBAS",  
  "fechaNacimiento": "1900-01-01"  
}
```

Response:

Code HTTP: 201 Created



```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "uuidRegistroOperacion": "79647742-f937-4dd2-d165-36405ddc715d",
  "fechaEfectivaOperacion": "2026-06-23T10:24:46.429Z",
  "estadoOperacion": "COMPLETA",
  "estadoJugador": "COD800",
  "datosLimitesJugador": [
    {
      "idDefinicionLimite": 1,
      "cuantiaActual": 70000,
      "tipoCambio": "BAJADA",
      "fechaInicio": "2026-06-23T04:00:00.000Z",
      "fechaFin": "2026-06-24T03:59:59.999Z",
      "acumuladoActual": 2700,
      "etiqueta": "Diario"
    },
    {
      "idDefinicionLimite": 2,
      "cuantiaActual": 175000,
      "tipoCambio": "BAJADA",
      "fechaInicio": "2026-06-23T04:00:00.000Z",
      "fechaFin": "2026-06-30T03:59:59.999Z",
      "acumuladoActual": 2700,
      "etiqueta": "Semanal"
    },
    {
      "idDefinicionLimite": 3,
      "cuantiaActual": 330000,
      "tipoCambio": "BAJADA",
      "fechaInicio": "2026-06-09T04:00:00.000Z",
      "fechaFin": "2026-07-07T03:59:59.999Z",
      "acumuladoActual": 2800,
      "etiqueta": "4 Semanas"
    }
  ]
}
```

1.1.1.1.22 Cancellation Response

September 2026



POST <https://aws-api-pre.ordenacionjuego.es/slhc-operadores/v1/anulacion>

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "idOperador": "1040",
  "identificadorJugador": "X0000000T",
  "uuidRegistroOperacion": "019ef402-d35d-7842-9723-55e5a2682fb5",
  "fechaEfectivaOperacion": "2026-06-18T00:00:00.000Z",
  "nombre": "PRUEBAS",
  "primerApellido": "ESPANOL",
  "segundoApellido": "NIEPRUEBAS",
  "fechaNacimiento": "1900-01-01"
}
```

Response:

Code HTTP: 200 OK

```
{
  "uuidOperacionOperador": "79647742-f937-4dd2-d165-36405ddc715d",
  "uuidRegistroOperacion": "019ef402-d67d-7842-9723-55e5a2682fb5",
  "datosLimitesJugador": [
    {
      "idDefinicionLimite": 1,
      "cuantiaActual": 70000,
      "tipoCambio": "BAJADA",
      "fechaInicio": "2026-06-23T04:00:00.000Z",
      "fechaFin": "2026-06-24T03:59:59.999Z",
      "acumuladoActual": 2200,
      "etiqueta": "Diario"
    },
    {
      "idDefinicionLimite": 2,
      "cuantiaActual": 175000,
      "tipoCambio": "BAJADA",
      "fechaInicio": "2026-06-23T04:00:00.000Z",
      "fechaFin": "2026-06-30T03:59:59.999Z",
      "acumuladoActual": 2200,
    }
  ]
}
```

September 2026



```
    "etiqueta": "Semanal"
  },
  {
    "idDefinicionLimite": 3,
    "cuantiaActual": 330000,
    "tipoCambio": "BAJADA",
    "fechaInicio": "2026-06-09T04:00:00.000Z",
    "fechaFin": "2026-07-07T03:59:59.999Z",
    "acumuladoActual": 2300,
    "etiqueta": "4 Semanas"
  }
]
}
```

7.3 Application Form for Registration or Modification in the Service

Gaming operators wishing to carry out tests with the Joint Deposit Limits Service must apply for registration in the test environment using the form published on the Electronic Office at “**Application for an access key to the Joint Deposit Limits System (SLDC)**” on the portal of the Directorate General for Gambling Regulation, ticking the “Pre-Production” checkbox.

Registration in the production environment will be requested using the same form and procedure, ticking the “Production” checkbox

The applicant company must be registered in the Special Section of Participants of the General Register of Gambling Licenses. Access to real data will be conditional on the granting of a specific license and on passing the tests in the pre-production environment.

Registration in both environments cannot be requested simultaneously in the same application. Each environment must have its own separate application. Access to the production environment will not be authorized until the tests in the PRE environment have been completed, in accordance with section 5.1.1.



Likewise, the request for the access key to the production environment will be conditional on the submission, in the access form for that environment, of evidence demonstrating that the system tests have been successfully carried out in the pre-production environment.

7.3.1 Request Data

This covers the data the operator must complete in order to submit the “**Application for an access key to the Joint Deposit Limits System (SLDC)**”

It consists of the following sections:

7.3.1.1 Types of Request

The information required in this application is equivalent to that which the operator usually provides in other existing application procedures, except as detailed below.

7.3.1.2 Environment

It will be validated that one of the two environments indicated in the “**Application for an access key to the Joint Deposit Limits System (SLDC)**”

Entorno

- ☐ Preproducción
- ☐ Producción

If both available types have been selected at the same time, the system will not allow the execution of the request.

7.3.1.3 Electronic Notification

Once the application has been submitted, a notification will be sent to the operator including an attached file with the corresponding access key. Access to that notification may only be made using the electronic certificate of the representative of the operator indicated in the application for notification purposes.