User Scenarios — Origo Cloud Events Landsbankinn

Publisher: Origo **Version:** 28.0.0.0 **Submission Date:** 2026-07-07 **Test Environment:** Requires Landsbankinn API test credentials. The extension connects to Landsbankinn's banking web services for statement retrieval, claim management, and payment operations. Provide test credentials valid for 4+ weeks.

Test Credentials

Item	Value
BC Environment	Sandbox with IS localization
Landsbankinn API URL	Provided by Landsbankinn (test environment)
Client Certificate	Test certificate for API authentication
Bank Account (kennitala)	Test company kennitala with active bank account
API User	Test user with full banking permissions

> ***Note:*** *All scenarios require a valid Landsbankinn API test environment. The extension communicates with Landsbankinn web services — no sandbox/mock mode exists. Contact Landsbankinn for test credentials.*

Scenario 1: Install and Configure Extension

Area: Installation & Setup

Setup

1. Start with a clean Business Central sandbox environment with IS localization.
2. Ensure "Origo Cloud Events Core" is already installed.

Steps

1. Install "Origo Cloud Events Landsbankinn" from AppSource.
2. Open the Cloud Events Setup page.
3. Verify that Landsbankinn message types appear in the Message Type list.
4. Configure the Landsbankinn API connection (certificate, endpoint URL, kennitala).

Expected Results

- Extension installs without errors.
- All Landsbankinn message types are registered and visible.
- Setup page accepts connection parameters.
- No existing functionality is disrupted.

Scenario 2: Retrieve Bank Statement

Area: Bank Statements

Setup

1. Extension is installed and configured with valid API credentials.

2. At least one bank account exists with recent transactions.

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Statement.Get`.
2. Include the bank account number and date range in the request payload.
3. Process the task via the Cloud Event Task API.
4. Retrieve the response from the Cloud Event Data API.

Expected Results

- Response contains bank statement data in structured JSON format.
- Statement includes transaction date, amount, reference, and balance fields.
- Date range filtering works correctly.
- Response content type is set to `application/json`.

Scenario 3: Query Claims

Area: Claims Management

Setup

1. Extension is configured with valid API credentials.
2. Test account has existing claims (kröfur) registered.

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Claim.Query`.
2. Include filter parameters (date range, status) in the request.
3. Process the task and retrieve the response.
4. Submit a second message with type `Landsbankinn.Claim.QueryOne` for a specific claim.

Expected Results

- `Claim.Query` returns a list of claims matching the filter criteria.
- `Claim.QueryOne` returns detailed information for a single claim.
- Claim data includes claim number, amount, due date, payer kennitala, and status.

Scenario 4: Query Claim Payments

Area: Claims Management

Setup

1. Extension is configured with valid API credentials.
2. Test account has claims with recorded payments.

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Claim.QueryPayments`.
2. Include the claim identifier in the request payload.
3. Process the task and retrieve the response.

Expected Results

- Response contains payment records associated with the specified claim.
- Payment data includes payment date, amount, payer information, and payment reference.

- Empty result set is returned for claims with no payments (not an error).

Scenario 5: Create and Manage Claim Batch

Area: Claims Management

Setup

1. Extension is configured with valid API credentials.
2. Prepare claim data (payer kennitala, amount, due date, reference).

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Claim.CreateBatch` containing multiple claims.
2. Note the batch ID from the response.
3. Submit a message with type `Landsbankinn.Claim.AlterBatch` to modify one claim in the batch.
4. Verify the alteration via `Landsbankinn.Claim.Query`.
5. Submit a message with type `Landsbankinn.Claim.CancelBatch` to cancel the remaining claims.

Expected Results

- `CreateBatch` returns a batch ID and confirmation of claims created.
- `AlterBatch` successfully modifies the specified claim.
- `CancelBatch` cancels claims and returns confirmation.
- Each operation returns appropriate status codes.

Scenario 6: Re-Create Claim Batch

Area: Claims Management

Setup

1. Extension is configured with valid API credentials.
2. A previously cancelled claim batch exists.

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Claim.ReCreateBatch`.
2. Reference the original batch ID in the request.
3. Process the task and retrieve the response.

Expected Results

- Previously cancelled claims are re-created with new batch ID.
- Original claim data (amounts, due dates, references) is preserved.
- Response confirms successful re-creation.

Scenario 7: Check Async Operation Status

Area: Operations

Setup

1. Extension is configured with valid API credentials.
2. A batch operation (create/alter/cancel) has been submitted.

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Claim.GetOperationResult`.
2. Include the operation ID from a previous batch submission.
3. Process the task and retrieve the response.

Expected Results

- Response indicates operation status (pending, completed, failed).
- For completed operations, detailed results are included.
- For failed operations, error details are provided.

Scenario 8: Submit Payment Batch

Area: Payments

Setup

1. Extension is configured with valid API credentials.
2. Prepare payment data (recipient kennitala, bank account, amount, reference).

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Payment.Batch` containing multiple payments.
2. Note the batch ID from the response.
3. Submit a message with type `Landsbankinn.Payment.ResultBatch` to retrieve results.

Expected Results

- `Payment.Batch` accepts the payment instructions and returns a batch ID.
- `Payment.ResultBatch` returns processing status for each payment in the batch.
- Individual payment statuses (accepted, rejected, pending) are clearly indicated.

Scenario 9: Foreign Payment Operations

Area: Payments

Setup

1. Extension is configured with valid API credentials.
2. Prepare foreign payment data (IBAN, SWIFT/BIC, amount, currency).

Steps

1. Submit a Cloud Event message with type `Landsbankinn.ForeignPayment.Create`.
2. Include recipient IBAN, SWIFT code, amount, and currency in the request.
3. Process the task and retrieve the response.
4. Submit a message with type `Landsbankinn.ForeignPayment.Query` to check status.

Expected Results

- `ForeignPayment.Create` accepts the payment instruction and returns a reference.
- `ForeignPayment.Query` returns status and details of foreign payment submissions.
- Currency and amount formatting is preserved correctly.

Scenario 10: Get Currency Rates

Area: Reference Data

Setup

1. Extension is configured with valid API credentials.

Steps

1. Submit a Cloud Event message with type `Landsbankinn.CurrencyRates.Get`.
2. Optionally include a specific date in the request.
3. Process the task and retrieve the response.

Expected Results

- Response contains exchange rates for supported currencies.
- Each rate includes currency code, buy rate, sell rate, and effective date.
- Rates are from Landsbankinn's official published rates.

Scenario 11: Verify Bank Account

Area: Reference Data

Setup

1. Extension is configured with valid API credentials.
2. Have a valid Icelandic bank account number (4-digit branch + 2-digit ledger + 6-digit account + check digit).

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Account.Verify`.
2. Include the bank account number in the request payload.
3. Process the task and retrieve the response.

Expected Results

- Valid account returns confirmation with account holder name.
- Invalid account returns a structured error indicating the account is not valid.
- Response clearly distinguishes between valid and invalid accounts.

Scenario 12: Query Unpaid Invoices and Payment Slips

Area: Queries

Setup

1. Extension is configured with valid API credentials.
2. Test account has unpaid invoices and payment slips.

Steps

1. Submit a Cloud Event message with type `Landsbankinn.UnpaidInvoice.Query`.
2. Process the task and retrieve the response.
3. Submit a message with type `Landsbankinn.PaymentSlip.Query`.

4. Process and retrieve that response.

Expected Results

- `UnpaidInvoice.Query` returns list of unpaid invoices with amounts and due dates.
- `PaymentSlip.Query` returns payment slip data.
- Both responses use consistent JSON structure.

Scenario 13: Error Handling — Invalid Credentials

Area: Error Handling

Setup

1. Extension is installed.
2. Configure with intentionally invalid API credentials (expired certificate or wrong kennitala).

Steps

1. Submit any Cloud Event message (e.g., `Landsbankinn.Statement.Get`).
2. Process the task.
3. Retrieve the response.

Expected Results

- Task does not crash or hang.
- Response contains a structured error message indicating authentication failure.
- Error message does not expose sensitive credential details.
- BC event log records the failure with a correlation ID.

Scenario 14: Error Handling — Service Unavailable

Area: Error Handling

Setup

1. Extension is configured (credentials valid but service endpoint unreachable — simulate by temporarily misconfiguring the URL).

Steps

1. Submit a Cloud Event message with type `Landsbankinn.Statement.Get`.
2. Process the task.
3. Retrieve the response.

Expected Results

- Task completes without crashing.
- Response contains a structured error indicating the service is unavailable.
- Error includes enough detail for troubleshooting (timeout, connection refused).
- No partial or corrupted data is written.

Scenario 15: Uninstall Extension

Area: Installation

Setup

1. Extension is installed and has been used (messages processed, configuration saved).

Steps

1. Uninstall "Origo Cloud Events Landsbankinn" from the environment.
2. Verify the system remains functional.
3. Confirm "Origo Cloud Events Core" still operates correctly for other message types.

Expected Results

- Extension uninstalls cleanly without errors.
- Landsbankinn message types are removed from the system.
- No orphaned data or broken references remain.
- Other Cloud Events message types continue to function.