Software Design Document (SDD) - PSCE Outbound Claim Extract

Document Version: 1.5 **Date:** October 7, 2025 **System:** Prime Claims Extract (PSCE) Outbound Pipeline

1 GENERAL INFORMATION

1.1 Executive Summary

This document defines the technical architecture and design specifications for the Prime Claims Extract (PSCE) Outbound pipeline. The primary objective of this pipeline is to securely process high-volume claims data originating from the Facets system, filter records that are in a 'Pended for PRIME Review' status, **enrich the data by fetching corresponding member details from a Member API**, and load the required data elements into a Snowflake data warehouse table for downstream processing by Prime Therapeutics. The solution is implemented using two decoupled, event-driven Azure Functions for high scalability and fault tolerance.

1.2 Scope

The PSCE Outbound pipeline focuses on moving, validating, filtering, **enriching**, and persisting claim data pointers and filtered results.

1.2.1 In-Scope

- **Function 1 (QueueToBlobFunction):** Reading raw message data from the Facets Outbound Queue and moving that raw message into a designated Azure Blob Storage container for durability and triggering.
- **Function 2 (BlobToSnowflakeFunction):** Triggered by the new blob, fetching the large claims JSON payload via a URL provided in the blob, validating the payload structure, applying the 'Pended' filtering business rule, **calling an external Member API for data enrichment**, merging the data, and performing bulk insertion into the Snowflake target table.
- **Logging & Auditing:** Implementation of detailed audit and error logging using Azure Application Insights (the designated Azure Error Log Service). **Only ERROR logs will be persisted to the centralized Snowflake Audit Log Table for operational review and retry management.** Successful audit logs will remain in Application Insights.
- **Target Database Schema:** Insertion of data into the DATA_MART.PSCE.PSCE_PENDED_CLAIMS_OUTBOUND Snowflake table.

1.2.2 Out-of-Scope

- The generation or creation of the initial message and data URL by the Facets system.
- The internal claims processing, review, and edits performed by Prime Therapeutics after

data is loaded into Snowflake.
- The final SFTP transfer of data *from* Snowflake to Prime.
- Facets-side configuration or adjudication logic that determines the initial 'Pended' status.

1.3 Document Usage

This document is intended for:
- **Developers:** To guide the implementation of Azure Functions, authentication, and data transformation logic.
- **Quality Assurance (QA):** To define test cases, focusing on input validation, business rule adherence (filtering logic), and **Member API integration**.
- **Support/Operations:** To understand the end-to-end process, dependencies, and to interpret audit and error logs in Application Insights, and to use the **Audit Log Table** for monitoring and manual/automated retry of failed records.

1.4 Diagram

1.4.1 Data Flow Diagram

Description: The process begins with a message (containing a data URL) placed in the Azure Queue (Source Queue) by the Facets system. Function 1 consumes this message and writes it to Azure Blob Storage (Metadata Container). This write triggers Function 2, which then uses the URL inside the metadata to pull the large claims JSON from the Claims Data Store. Function 2 filters the claims, **calls the Member API for each pended claim**, merges the member data, and loads the qualifying records into Snowflake. Application Insights captures all audit and error logs from both functions, and **only ERROR logs are persisted to the new Audit Log Table in Snowflake.**

1.4.2 Use Case Diagram

Description:
- **Actors:** Facets System, Azure Functions Pipeline, **Member API**, Snowflake DB, Operations Team (monitoring).
- **Use Cases:** Receive Claim Pointer, Store Pointer Metadata, Trigger Claim Processing, Fetch Large Claims JSON, Filter Pended Claims, **Enrich Claim with Member Data (New)**, Insert into Snowflake, Log Operational Events (Audit/Error), Track and Manage Failed Transactions.

1.5 Specification Sheet

Component	Type	Trigger	Purpose	Output/Target
Function 1	Azure Function (Queue Trigger)	Azure Storage Queue (facets-outbound-queue)	Persist message metadata.	Azure Blob Storage (psce-metadata-container), **Snowflake Audit Log (on error**

Component	Type	Trigger	Purpose	Output/Target
				only)
Function 2	Azure Function (Blob Trigger)	Azure Blob Storage (psce-metadata-container)	Fetch, Filter, **Call Member API for enrichment**, Transform, Load.	Snowflake DB (PSCE_PENDED_CLAIMS_OUTBOUND), **Snowflake Audit Log (on error only)**
Logging	Azure Service	Synchronous/Asynchronous calls from F1 & F2	Capture Audit/Error Logs.	Azure Application Insights, **Snowflake Audit Log (Critical Errors Only)**

2 TECHNICAL DESIGN

2.1 Input for Claim Details

The process is initiated by the Facets system placing a message into the Source Queue. This message is not the full claims data, but a pointer to the location of the large data file.

2.1.1 Input JSON Structure

A. Queue Message (Pointer/Metadata - Input to F1): This is the raw message consumed by Function 1 and written to the Blob.

```
{
    "messageId": "FACETS-MSG-20251007-001234",
    "timestamp": "2025-10-07T14:30:00Z",
    "dataUrl":
"[https://claimsdatastore.azurewebsites.net/api/claimsextract/20251007
/001234.json](https://claimsdatastore.azurewebsites.net/api/claimsextr
act/20251007/001234.json)",
    "sourceSystem": "Facets"
}
```

B. Large Claims JSON Payload (Fetched by F2 via dataUrl): This is the full claims payload fetched from the dataUrl.

```
{
    "extractId": "001234",
    "claimsBatch": [
        {
            "claimId": "C001001A",
            "claimLineNumber": 1,
            "facetsStatus": "Pend for PRIME Review",
            "memberId": "M00987",
            "NDC": "00000000101",
            "quantityDispensed": 50.0,
            "unitOfMeasure": "ML",
            "primaryDiagnosis": "J45.909",
            "procedureCode": "J9001",
            "prescribingNPI": "1234567890",
            "renderingNPI": "0987654321",
            "serviceDate": "2025-09-01",
            "totalCharges": 1500.50,
            "groupId": "G100"
        },
        // ... many more claim records, including non-pended statuses
    ]
}
```

2.1.2 Input Request Details

1. **Function 1:** Triggered directly by the message arriving in the Azure Queue. Authentication is handled by the Function's managed identity or connection string.
2. **Function 2:**
    - Triggered by the blob write operation from Function 1.
    - Performs an **authenticated HTTP GET request** to the dataUrl (e.g., using a system-assigned identity or a pre-shared key) to retrieve the large claims JSON payload.

2.1.3 Input - Request Validation

Function 1 Validation (Queue Message):
- Check for existence and non-empty values for messageId and dataUrl.

Function 2 Validation (Claims JSON Payload):
- Ensure the fetched JSON is valid (JSON parsing successful).
- Verify that the root element contains the claimsBatch array.
- For each claim in claimsBatch, validate the presence of the mandatory fields required by the Snowflake schema (e.g., claimId, claimLineNumber, facetsStatus).

2.1.4 Member API Interaction (New)

Function 2 will execute an authenticated HTTP GET request to the Member API for every claim line that passes the Pended status filter (Rule 3.1.1).

A. Member API Request (Input to API):
- **Endpoint:** [MEMBER_API_URL]/api/members/{memberId}
- **Parameter:** memberId (sourced from claimsBatch[i].memberId)
- **Authentication:** Must use an Azure Managed Identity or Service Principal for secure access.

B. Member API Response (Output from API):
- **HTTP Status:** Expected 200 OK for success.
- **Payload Example:**

<!-- end list -->

```
{
    "memberId": "M00987",
    "firstName": "Jane",
    "lastName": "Doe",
    "dateOfBirth": "1990-01-15",
    "memberStatus": "Active"
}
```

2.2 Output for Claim Details

The primary output of the entire process is the structured data record inserted into the Snowflake table, now inclusive of Member API data.

2.2.1 Output JSON Structure (Internal Transformation)

After filtering, **enrichment**, and transformation, the internal record structure (prior to insertion) will strictly adhere to the Snowflake table schema definition.

2.2.2 Output Data Transformation (Header/Root Fields)

Source Field (Claims JSON)	Target Snowflake Column	Data Type	Transformation Logic
claimsBatch[i].claimId	CLAIM_ID	VARCHAR(50)	Direct mapping.
claimsBatch[i].claimLineNumber	CLAIM_LINE_NUMBER	INTEGER	Direct mapping.
claimsBatch[i].memberId	MEMBER_ID	VARCHAR(50)	Direct mapping.
claimsBatch[i].groupId	GROUP_ID	VARCHAR(50)	Direct mapping.
Calculated	PRIME_REFERENCE_ID	VARCHAR(100)	Generate unique ID if required by Prime (e.g., CLAIM_ID + LINE_NUMBER + EXTRACT_DATE).

Source Field (Member API)	**Target Snowflake Column**	**Data Type**	**Transformation Logic**
firstName	**MEMBER_FIRST_NAME**	**VARCHAR(50)**	**Direct mapping.**
lastName	**MEMBER_LAST_NAME**	**VARCHAR(50)**	**Direct mapping.**
dateOfBirth	**MEMBER_DOB**	**DATE**	**Direct mapping and date formatting.**
memberStatus	**MEMBER_STATUS**	**VARCHAR(20)**	**Direct mapping.**

2.2.3 Output Data Transformation (Medical Line Fields)

Source Field (Claims JSON)	Target Snowflake Column	Data Type	Transformation Logic
claimsBatch[i].NDC	NDC_CODE	VARCHAR(11)	Direct mapping.

Source Field (Claims JSON)	Target Snowflake Column	Data Type	Transformation Logic
claimsBatch[i].quantityDispensed	QUANTITY_DISPENSED	NUMERIC(18, 2)	Direct mapping.
claimsBatch[i].unitOfMeasure	QUANTITY_UNIT_OF_MEASURE	VARCHAR(10)	Direct mapping.
claimsBatch[i].primaryDiagnosis	DIAGNOSIS_CODE_PRIMARY	VARCHAR(20)	Direct mapping (ICD-10).
claimsBatch[i].procedureCode	PROCEDURE_CODE	VARCHAR(20)	Direct mapping (CPT/HCPCS).
claimsBatch[i].prescribingNPI	PRESCRIBING_PROVIDER_NPI	VARCHAR(20)	Direct mapping.
claimsBatch[i].renderingNPI	RENDERING_PROVIDER_NPI	VARCHAR(20)	Direct mapping.
claimsBatch[i].facetsStatus	FACETS_PEND_STATUS	VARCHAR(50)	Direct mapping.
claimsBatch[i].serviceDate	SERVICE_DATE	DATE	Direct mapping and date formatting.
claimsBatch[i].totalCharges	TOTAL_CHARGES	NUMERIC(18, 2)	Direct mapping.
Calculated	PSCE_EXTRACT_DATETIME	TIMESTAMP_NTZ(9)	F2 Function execution timestamp.

2.3 Audit and Error Logging Table

To enable comprehensive error tracking and a **retry mechanism** for failed claims, the DATA_MART.PSCE.PSCE_AUDIT_ERROR_LOG table will **exclusively** capture records where a non-recoverable error occurred (e.g., F1002, F2004) or a key failure milestone was reached. Successful transactions (AUDIT logs) are primarily recorded in Application Insights.

Logging Context Determination:

The DATA_URL, CLAIM_ID, and CLAIM_LINE_NUMBER fields are critical for determining the retry scope. Their population rules are as follows:

1. **Function 1 (Batch/Process Errors - F100X):** Errors occurring here affect the entire batch (e.g., failed to read queue message, failed to write blob).
   - **DATA_URL** will be populated (if available from the queue message).
   - **CLAIM_ID** and **CLAIM_LINE_NUMBER** will be set to **NULL**. The retry mechanism must re-initiate the process for the entire batch URL.
2. **Function 2 (Claim/Line Errors - F2004, F2005):** Errors occurring here affect specific claim records after the payload has been fetched and parsed.
   - **DATA_URL** will be populated.
   - **CLAIM_ID** and **CLAIM_LINE_NUMBER** will be populated with the details of the failing record. The retry mechanism can target this specific claim line.

Retry Flow:

1. **Identification:** Operations/Support team queries the PSCE_AUDIT_ERROR_LOG table for records where LOG_TYPE = 'ERROR', IS_RESOLVED = FALSE, and RETRY_COUNT < 3 (configurable limit).
2. **Correction:** The root cause (e.g., a data issue, a temporary network failure) is investigated and corrected externally.

3. **Trigger:** An automated process or a manual tool reads the DATA_URL and CLAIM_ID from the failed log record and re-triggers the Function 2 logic for that specific claim or batch.
4. **Update Log:** Upon successful re-processing, the IS_RESOLVED flag is set to TRUE and RETRY_COUNT is incremented.

3 BUSINESS RULES

3.1 Claim Filtering Logic (Prime-Pended Claims)

Function 2 is responsible for applying the core business filter to the fetched claims payload.
Rule 3.1.1 (Primary Status Filter):

- **Condition:** Only claim lines where the source field claimsBatch[i].facetsStatus is an exact match for the configured Pended Status.
- **Target Value (as per SRS):** 'Pend for PRIME Review'
- *Action:* If status matches, the record is flagged for transformation and insertion into Snowflake. If not, the record is skipped, and a low-level audit log entry is made (CLAIM_SKIPPED, reason: Status Not Pended) **only in Application Insights.** No entry is written to the Snowflake Audit Log Table.

Rule 3.1.2 (Mandatory Data Check):

- **Condition:** All Pended claims must contain non-null and non-empty values for: NDC_CODE, QUANTITY_DISPENSED, DIAGNOSIS_CODE_PRIMARY, and PROCEDURE_CODE.
- *Action:* If a Pended claim fails this check, it is **not** inserted into Snowflake. An **ERROR** log entry is created (F2004: MISSING_MANDATORY_DATA), detailing the missing fields and the CLAIM_ID. **This ERROR log is written to both Application Insights and the Audit Log Table, with IS_RESOLVED set to FALSE.**

Rule 3.1.3 (Member Data Enrichment):

- **Condition:** For every claim line successfully filtered by Rule 3.1.1, Function 2 must successfully retrieve member details from the Member API.
- *Action:* If the Member API returns an HTTP status code other than 200, or if the returned payload is invalid/missing critical fields, the claim line is **not** inserted into Snowflake. An **ERROR** log entry is created (F2006: MEMBER_API_FETCH_FAILURE), detailing the API call error. **This ERROR log is written to both Application Insights and the Audit Log Table, with IS_RESOLVED set to FALSE.**

3.2 Error Handling & Logging

All logging will utilize **Azure Application Insights** for structured, centralized error and audit tracking, satisfying the 'Azure Error Log Service' requirement. Critical logs (**all errors**) will be persisted to the PSCE_AUDIT_ERROR_LOG Snowflake table. Successful batch-level and record-level AUDIT logs will reside solely in Application Insights.
A. Audit Logging (Type: AUDIT): Used for successful operational steps to track progress and performance. **These logs are written ONLY to Azure Application Insights.** | Field | Value | Example Event | | :--- | :--- | :--- | | LogType | AUDIT | Function start, Blob write success, X claims filtered, Snowflake insert success. | | ProcessStep | BLOB_WRITE_SUCCESS, CLAIM_FILTERED, SNOWFLAKE_INSERT | | | CorrelationId |

FACETS-MSG-20251007-001234 | The queue/blob message ID. | | RecordsProcessed | 4500 | Count of claims processed/filtered/inserted. |

B. Error Logging (Type: ERROR): Used for process failures requiring investigation and potential manual intervention. Each error entry **must** include a custom ErrorCode. Errors are written to both Application Insights and **are persisted to the Audit Log Table.**

Error Code	Function	Description	Retry Context Fields Populated
F1001	F1	Queue Message Read Failure.	DATA_URL (if available); CLAIM_ID=NULL; CLAIM_LINE_NUMBER=NULL
F1002	F1	Blob Write/Connection Failure.	DATA_URL (from input); CLAIM_ID=NULL; CLAIM_LINE_NUMBER=NULL
F2001	F2	Input JSON (Metadata) Validation Error.	DATA_URL; CLAIM_ID=NULL; CLAIM_LINE_NUMBER=NULL
F2002	F2	Data URL Fetch Failure (HTTP Error).	DATA_URL; CLAIM_ID=NULL; CLAIM_LINE_NUMBER=NULL
F2003	F2	Claims JSON Payload Structure Invalid (Parsing error).	DATA_URL; CLAIM_ID=NULL; CLAIM_LINE_NUMBER=NULL
F2004	F2	Missing Mandatory Data (Rule 3.1.2).	DATA_URL, CLAIM_ID, CLAIM_LINE_NUMBER
F2005	F2	Snowflake Insertion/SQL Error.	DATA_URL, CLAIM_ID, CLAIM_LINE_NUMBER (of the specific record that failed)
F2006	**F2**	**Member API Fetch Failure (Rule 3.1.3).**	**DATA_URL, CLAIM_ID, CLAIM_LINE_NUMBER**

4 REFERENCE

PSCE_PENDED_CLAIMS_OUTBOUND (Snowflake Table DDL - Target Data)

```
CREATE TABLE DATA_MART.PSCE.PSCE_PENDED_CLAIMS_OUTBOUND (
    -- Unique Key & Identifiers
    CLAIM_ID                        VARCHAR(50)     NOT NULL,
    CLAIM_LINE_NUMBER               INTEGER         NOT NULL,
    MEMBER_ID                       VARCHAR(50),
    GROUP_ID                        VARCHAR(50),
    -- Member API Enrichment Fields (NEW)
    MEMBER_FIRST_NAME               VARCHAR(50),
    MEMBER_LAST_NAME                VARCHAR(50),
    MEMBER_DOB                      DATE,
    MEMBER_STATUS                   VARCHAR(20),
    PRIME_REFERENCE_ID              VARCHAR(100),

    -- CRITICAL ADDITIONS (Medical Pharmacy Data Elements for Edits)
    NDC_CODE                        VARCHAR(11),
    QUANTITY_DISPENSED              NUMERIC(18, 2),
    QUANTITY_UNIT_OF_MEASURE        VARCHAR(10),
    DIAGNOSIS_CODE_PRIMARY          VARCHAR(20),
    PROCEDURE_CODE                  VARCHAR(20),
    PRESCRIBING_PROVIDER_NPI        VARCHAR(20),
    RENDERING_PROVIDER_NPI          VARCHAR(20),

    -- Service & Pended Status Details (Original Fields)
    FACETS_PEND_STATUS              VARCHAR(50)     NOT NULL,
    SERVICE_DATE                    DATE,
    BILL_TYPE                       VARCHAR(10),
    REVENUE_CODE                    VARCHAR(10),
    TOTAL_CHARGES                   NUMERIC(18, 2),

    -- Audit and Processing Metadata
    PSCE_EXTRACT_DATETIME           TIMESTAMP_NTZ(9) NOT NULL,
    AZURE_BLOB_SOURCE_URL           VARCHAR(500)    NOT NULL,
    AZURE_FUNCTION_RUN_ID           VARCHAR(100)    NOT NULL,
    SNOWFLAKE_INSERT_DATETIME       TIMESTAMP_NTZ(9) DEFAULT
CURRENT_TIMESTAMP(),

    -- Constraints
    PRIMARY KEY (CLAIM_ID, CLAIM_LINE_NUMBER)
);
```

PSCE_AUDIT_ERROR_LOG (Snowflake Table DDL - Transaction Log and Retry)

This table captures all critical events for monitoring and provides the necessary context for re-triggering failed transactions.

```
CREATE TABLE DATA_MART.PSCE.PSCE_AUDIT_ERROR_LOG (
    -- Primary Key and Timing
    LOG_ID                          VARCHAR(36)     NOT NULL PRIMARY
KEY, -- UUID/GUID for PK
    LOG_DATETIME                    TIMESTAMP_NTZ(9) DEFAULT
CURRENT_TIMESTAMP(),

    -- Process Identification
    LOG_TYPE                        VARCHAR(10)     NOT NULL, -- Must
be 'ERROR' in this table.
    AZURE_FUNCTION                  VARCHAR(5)      NOT NULL, -- 'F1'
or 'F2'
    CORRELATION_ID                  VARCHAR(100)    NOT NULL, --
Facets Message ID (Batch/Group Identifier)
    AZURE_FUNCTION_RUN_ID           VARCHAR(100)    NOT NULL,

    -- Event/Error Details
    PROCESS_STEP                    VARCHAR(50)     NOT NULL,
    ERROR_CODE                      VARCHAR(10),    -- Custom Error
Code (e.g., F2004).
    MESSAGE                         VARCHAR(MAX),   -- Detailed log
message / Exception stack trace

    -- Retry Context (Used primarily for ERROR logs)
    DATA_URL                        VARCHAR(500),   -- URL to the full
claim payload (for F2 retries). Populated by both F1 (Batch errors)
and F2 (All errors).
    CLAIM_ID                        VARCHAR(50),    -- Specific Claim
ID that failed (if known). NULL for F1 errors and F2 batch-level
errors.
    CLAIM_LINE_NUMBER               INTEGER,        -- Specific Claim
Line that failed (if known). NULL for F1 errors and F2 batch-level
errors.
    RETRY_COUNT                     INTEGER         DEFAULT 0,
    IS_RESOLVED                     BOOLEAN         DEFAULT FALSE, --
Set to TRUE upon successful reprocessing

    -- Raw Data (For deep investigation of F2004 type errors)
    RAW_INPUT_DATA                  VARCHAR(MAX)    -- Small chunk of
input data relevant to the failure (e.g., the specific claim JSON
object that failed validation)
);
```

5 AMENDMENT HISTORY

Version	Date	Author	Description of Change
1.0	2025-10-07	Gemini LLM	Initial SDD creation based on SRS

Version	Date	Author	Description of Change
			requirements, defining the Azure Function pipeline, data flow, logging strategy, and target Snowflake schema.
1.1	2025-10-07	Gemini LLM	Added section 2.3 for the Audit and Error Logging Table and included the full DDL for PSCE_AUDIT_ERROR_LOG to support transaction tracking and the claim retry mechanism.
1.2	2025-10-07	Gemini LLM	Updated logging strategy to ensure that the Snowflake Audit Log Table only captures ERROR logs for retriability, and not routine successful AUDIT transactions, based on user feedback.
1.3	2025-10-07	Gemini LLM	Clarified logging rules in Section 2.3 and 3.2.B detailing when Function 1 vs Function 2 populates DATA_URL, CLAIM_ID, and CLAIM_LINE_NUMBER to support batch-level vs. claim-level retries.
1.4	2025-10-07	Gemini LLM	Updated Function 2 logic to include an enrichment step: calling an external Member API using memberId, merging the response with claim data, and inserting the combined record. Added Rule 3.1.3, Error Code

Version	Date	Author	Description of Change
			F2006, new fields in Section 2.2, and updated the target DDL accordingly.
1.5	**2025-10-07**	**Gemini LLM**	**Replaced the brevity placeholder in Section 4 with the full DDL script for PSCE_AUDIT_ERROR_LOG. Also corrected formatting errors in the data transformation table in Section 2.2.2.**