Software Design Document (SDD) - PSCE Outbound Claim Extract

Document Version: 1.0 **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, 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, 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, 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).
- **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 and business rule adherence (filtering logic).
- **Support/Operations:** To understand the end-to-end process, dependencies, and to interpret audit and error logs in Application Insights.

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 and loads the qualifying records into Snowflake. Application Insights captures all audit and error logs from both functions.

1.4.2 Use Case Diagram

Description:

- **Actors:** Facets System, Azure Functions Pipeline, Snowflake DB, Operations Team (monitoring).
- **Use Cases:** Receive Claim Pointer, Store Pointer Metadata, Trigger Claim Processing, Fetch Large Claims JSON, Filter Pended Claims, Insert into Snowflake, Log Operational Events (Audit/Error).

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)
Function 2	Azure Function (Blob Trigger)	Azure Blob Storage (psce-metadata-container)	Fetch, Filter, Transform, Load.	Snowflake DB (PSCE_PENDED_CLAIMS_OUTBOUND)
Logging	Azure Service	Synchronous/Asynchronous calls from F1 & F2	Capture Audit/Error Logs.	Azure Application Insights

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).

2.2 Output for Claim Details

The primary output of the entire process is the structured data record inserted into the Snowflake table.

2.2.1 Output JSON Structure (Internal Transformation)

After filtering and transformation, the internal record structure (prior to insertion) will strictly adhere to the Snowflake table schema definition provided in the reference section.
Mandatory Fields: CLAIM_ID, CLAIM_LINE_NUMBER, FACETS_PEND_STATUS, AZURE_BLOB_SOURCE_URL, AZURE_FUNCTION_RUN_ID, PSCE_EXTRACT_DATETIME.

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).

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.
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.

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).

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.

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.
A. Audit Logging (Type: AUDIT): Used for successful operational steps to track progress and performance. | 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.

Error Code	Function	Description	Action/Mitigation
F1001	F1	Queue Message Read Failure.	Log raw message data; investigate Queue format.
F1002	F1	Blob Write/Connection Failure.	Log Blob path and exception details; investigate Blob permissions.
F2001	F2	Input JSON (Metadata) Validation Error.	Log malformed JSON string; investigate Facets output format.
F2002	F2	Data URL Fetch Failure (HTTP Error).	Log dataUrl and HTTP status code (404, 401, timeout).
F2003	F2	Claims JSON Payload Structure Invalid.	Log root error; check for missing claimsBatch key.
F2004	F2	Missing Mandatory Data (Rule 3.1.2).	Log CLAIM_ID and the specific missing data fields.
F2005	F2	Snowflake Insertion/SQL Error.	Log the batch of failed records and the full Snowflake exception/error code.

4 REFERENCE

The final, approved table definition for the claims data storage:

~~~
### PSCE_PENDED_CLAIMS_OUTBOUND (Snowflake Table DDL)

```sql
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),
    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) NOT DEFAULT
CURRENT_TIMESTAMP(),

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

## 5 AMENDMENT HISTORY

Version	Date	Author	Description of Change
1.0	2025-10-07	Gemini LLM	Initial SDD creation based on SRS requirements, defining the Azure Function pipeline, data flow, logging strategy, and target Snowflake schema.