**High-Level Design**

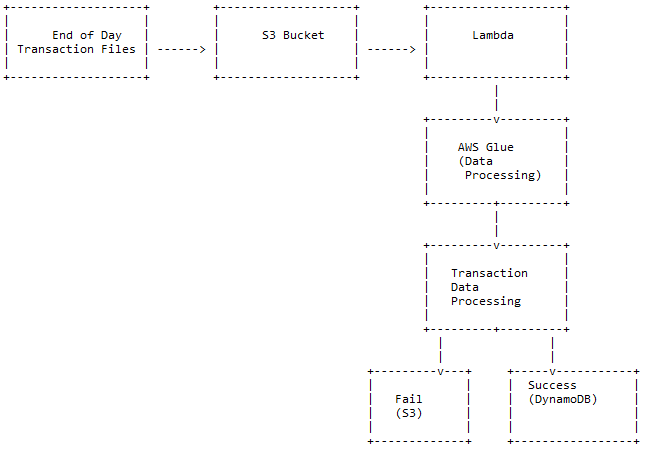
**Project 2**

**Upload EOD Transactions files into System**

**Infrastructure Overview:**

* AWS S3: Used to store incoming transaction files.
* AWS Lambda: Responsible for processing and posting transactions.
* DynamoDB: Stores the Account Master and Source System Master data.
* AWS Glue: Optional, can be used for data transformations if required.

**Infrastructure Setup Block Diagram:**

****

**System Workflow:**

* The system will receive two types of files: Parquet and Avro.
* Each file contains transaction data from a satellite system.
* The files will be uploaded to an S3 bucket.
* AWS Lambda will be triggered by the arrival of new files in the S3 bucket.
* Lambda will read the file, process the data, and generate ledger transactions.
* For each transaction, three separate entries will be posted to the Transaction table.
* Transaction amounts will be credited to the appropriate accounts (e.g., "Product Sale," "Value Added Tax," "Excise Duty," "Custom Duty").
* If a transaction fails, a separate file containing failed records will be created for reloading purposes.

**Account Master Table (DynamoDB):**

* Table Name: acct\_master
* Columns: acc\_no (Text, Unique alphanumeric), acc\_name (Text), acc\_desc (Text), acc\_type (Text, values: I, E, A, L)

**Transaction Table (DynamoDB):**

* Table Name: ledger\_txn
* Columns: txn\_id (Integer, sequential auto-generated), voucher\_code (Char(10), unique), txn\_type (Char(1), values: D, C),
* txn\_date (Date), acc\_no (Foreign Key from acct\_master), txn\_amt (Numeric(11.2)), source\_system\_id (Integer, values: 1, 2),
* source\_system\_txn\_id (Varchar(50))

**Source System Master Table (DynamoDB):**

* Table Name: src\_sys\_mst
* Columns: system\_id (Integer, sequential auto-generated), system\_name (Varchar(30))

**Data Processing Block Diagram:**

**A picture containing diagram, text, line, plan

Description automatically generated**

**Processing Steps:**

* For Parquet Files:
* Read the Parquet file from the S3 bucket.
* Extract the necessary fields (trnrefid, code, tdate, trn\_amount, vat, excise\_duty).
* Generate voucher code from trnrefid.
* Create three separate transaction entries for each transaction:
* Credit "Product Sale" with trn\_amount.
* Credit "Value Added Tax" with vat.
* Credit "Excise Duty" with excise\_duty.
* Validate the transaction date to ensure it's not a future date.
* Post the transactions to the ledger\_txn table.
* If any transaction fails, create a separate file for failed records.
* For Avro Files:
* Read the Avro file from the S3 bucket.
* Extract the necessary fields (Tran\_ref\_id, Transaction\_dt, amt, gst, custom\_duty).
* Generate voucher code from Tran\_ref\_id.
* Create three separate transaction entries for each transaction:
* Credit "Product Sale" with amt.
* Credit an appropriate account (not specified in the requirements) with gst.
* Credit "Custom Duty" with custom\_duty.
* Validate the transaction date to ensure it's not a future date.
* Post the transactions to the ledger\_txn table.
* If any transaction fails, create a separate file for failed records.

**Automation Considerations:**

* Set up AWS CloudWatch Events to trigger the Lambda function at the end of each day when new files arrive in the S3 bucket.
* Configure Lambda to handle file processing and posting of transactions automatically.
* Schedule periodic checks for failed transactions and reload the data if needed.

This HLD provides a high-level overview of the system design based on the given requirements. It's important to consider additional factors, such as security, performance optimization, and scalability, when implementing the system in a production environment.