HIGH LEVEL DOCUMENTATION -2

PROJECT-2

**Project Title: Upload EOD Transactions files into System**

**AWS Services:**

* 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

**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)

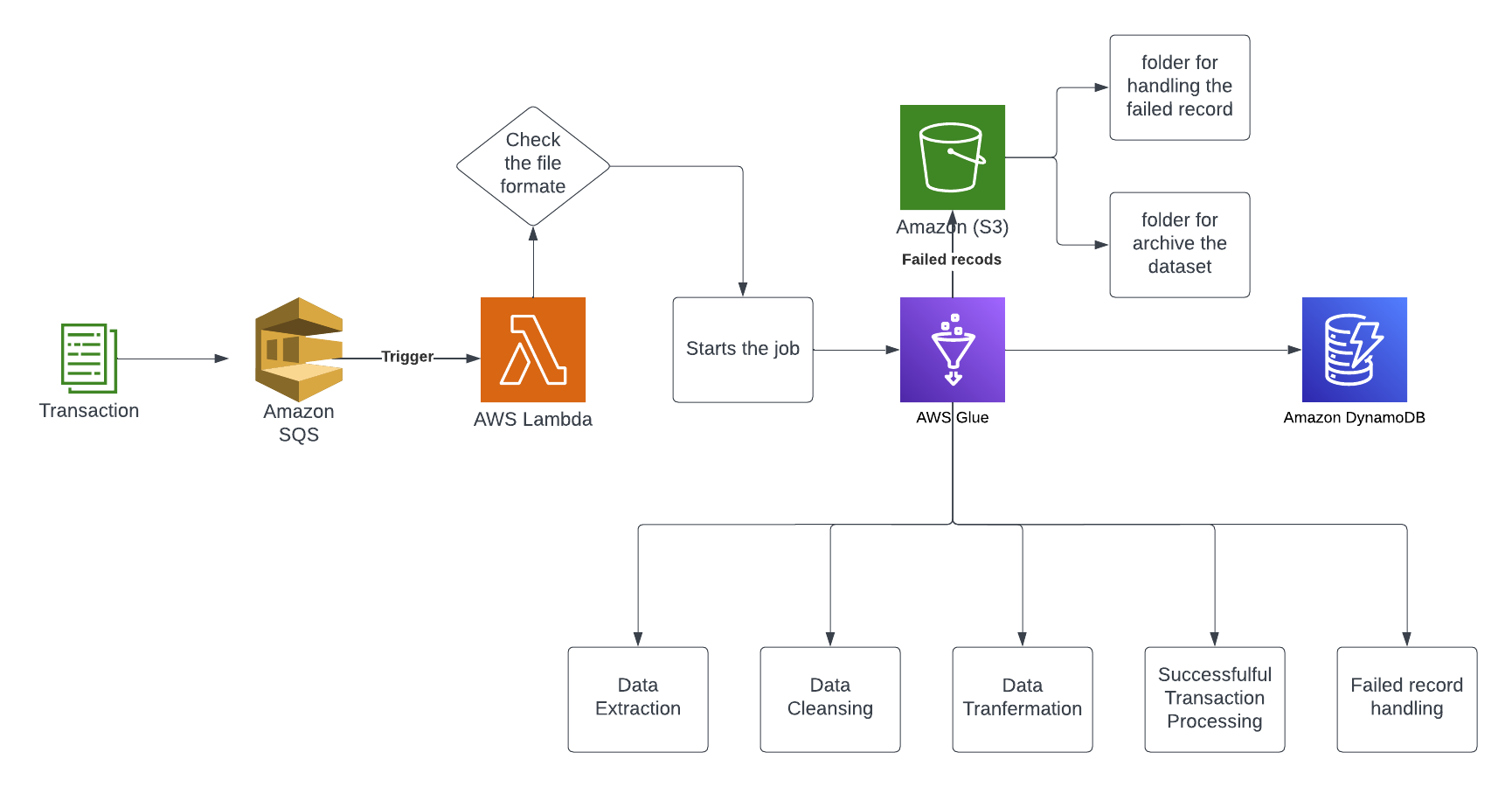
**Transaction Table (DynamoDB):**

* 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:**



**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.

## **Architectural Plan**

## The architecture follows a distributed and scalable approach to handle the transaction data. It consists of the following key components:

**Data Loading Component**

* Identify and integrate with transaction data sources, such as external APIs or data providers.
* Implement a data loading process to extract data from the sources and transform it into a suitable format for processing.
* Leverage AWS Glue jobs or custom scripts to retrieve and transform the data.
* Load the transformed data into Amazon S3 for storage and further processing.

**Data Storage Component**

* Utilize Amazon S3 as the primary data storage solution.
* Organize the transaction data in structured formats to optimize data retrieval and processing.

**Data Processing Component**

* Utilize AWS Glue for data processing and transformation.
* Develop AWS Glue jobs or ETL scripts to apply transformations, data cleansing, filtering, aggregation, or enrichment as required.
* Implement data quality checks to ensure the accuracy, completeness, and consistency of the processed data.

## **Transactions**

**Transactions**: The different types of transactions or operations that users can perform within the system. This can include:

* **Data Loading:** This is the transaction flow for loading transaction data into the system. This involves steps such as data source integration, data extraction, and loading the data into Amazon S3.
* **Data Processing:** This is the transaction flow for processing the transaction data. This involves steps such as transformation and ETL processes, and data quality checks.
* **Store Processed Data:** This is the transaction flow for users to retrieve or access the processed transaction data. This involves storing the processed data into DynamoDB table.

# **Out-of-scope**

# The project does not involve the development of a user interface or visualization components. The focus is on backend data storage and processing, and any frontend or user interface-related development is considered out of scope.

* The project is limited to transaction data storage and processing. Integrating external data sources beyond the scope of sales transaction, such as weather data or player social media feeds, is considered out of scope.