**Deliverables**

**1. Approach**

**Design and Implementation**

**ETL Pipeline Design**:

* **Extract**:
  + Use PySpark to read data from CSV and XLSX files.
  + CSV:

A screenshot of a computer program

Description automatically generated

* + Excel:

A screenshot of a computer program

Description automatically generated

* + Implement Spark DataFrame readers that handle different formats and structures.

A screenshot of a computer program

Description automatically generated

* **Transform**:
  + Clean and preprocess data to handle inconsistencies, such as varying company names.
  + Use fuzzy matching techniques (fuzzywuzzy library) to standardize company names.
  + Generate unique IDs where missing (especially for companies in the XLSX dataset).
  + Transform the data into a unified schema.

A screenshot of a computer

Description automatically generated

* **Load**:
  + Use Spark to handle database connections and operations.
  + Load the transformed data into PostgreSQL, maintaining metadata for traceability.

Companies:

A screenshot of a computer

Description automatically generated

Assets:

A screenshot of a table

Description automatically generated

Indicators:

A screenshot of a computer

Description automatically generated

Metadata

A screenshot of a data source

Description automatically generated

**Tools and Methodologies**:

* **PySpark**: For its distributed data processing capabilities and ease of integration with various data sources.
* **PostgreSQL**: As a robust and widely-used open-source relational database.
* **Docker**: To containerize the services for consistency across environments.
* **GitHub**: For version control and CI/CD integration.

**2. Data Quality and Traceability**

**Strategies for Data Quality**:

* Implement data validation checks using PySpark to ensure data integrity before loading.
* Use PostgreSQL constraints and triggers to enforce data quality rules.

**Data Lineage Tracking**:

* Maintain metadata columns (e.g., source\_file, load\_timestamp) in PostgreSQL tables.
* Log ETL operations to provide a traceable history of data transformations and loads.

**Error Correction Mechanisms**:

* Implement try-except blocks to handle and log errors during ETL processes.
* Use custom error handling functions to correct common issues (e.g., missing values, type mismatches).

**3. Data Integration**

**Approach to Data Integration**:

* Use PySpark to normalize data structures from different sources.
* Apply fuzzy matching to standardize and merge company names.
* Create a mapping table in PostgreSQL to link variations of company names to a unique identifier.
* Implement a unified schema that incorporates all necessary fields from both datasets.

**Time Estimate**

**Phase 1: Setup and Initial Configuration (1 day)**

* **Install and configure tools:** Install Python, PySpark, PostgreSQL, Docker, and other required libraries.
* **Setup version control:** Initialize a GitHub repository and set up the project structure.

**Phase 2: Data Extraction (1 day)**

* **Read CSV data:** Implement PySpark code to read the CSV file.
* **Read XLSX data:** Implement PySpark code to read the Excel file.

**Phase 3: Data Transformation (2 days)**

* **Standardize company names:** Implement functions to clean and standardize company names.
* **Generate unique IDs:** Create logic to generate unique IDs for companies lacking them in the XLSX dataset.
* **Transform to unified schema:** Map and transform the data into a unified schema.

**Phase 4: Data Loading (1 day)**

* **Setup PostgreSQL schema:** Define and create tables in PostgreSQL.
* **Load data into PostgreSQL:** Implement logic to load the transformed data into the PostgreSQL database.

**Phase 5: Data Quality and Traceability (1 day)**

* **Data validation checks:** Implement validation checks to ensure data integrity.
* **Metadata tracking:** Add metadata columns and implement logging for ETL operations.

**Phase 6: Testing and Error Handling (1 day)**

* **Unit tests:** Write unit tests for ETL functions.
* **Error handling:** Implement try-except blocks and custom error handling functions.

**Phase 7: Documentation and Finalization (1 day)**

* **Documentation:** Document the ETL process, including code comments and a detailed README file.
* **Review and refine:** Conduct a final review, perform refinements, and ensure all components work seamlessly.

**Total Estimated Time: 7 days**

**Additional details:**

 **CI/CD Implementation:** For CI/CD, Jenkins, Docker, and Kubernetes can be used to run Docker and PySpark. Alternatively, AWS Glue, Databricks, or EMR can be used for more robust ETL processes depending on data size and complexity.

 **Orchestration and Monitoring:** Apache Airflow can be used for orchestrating, monitoring, and alerting ETL workflows. AWS CloudWatch, SQS, and SNS can be integrated for additional monitoring and alerting capabilities.