### **1. Approach**

#### **Design and Implementation of the ETL Pipeline:**

1. **Technology Stack**:
   * **Python**: Core programming language for the ETL pipeline.
   * **Apache Spark (PySpark)**: For scalable data processing.
   * **PostgreSQL**: Data warehouse.
   * **Pandas**: For initial data ingestion if necessary.
   * **GitHub**: Version control for code and configurations.
   * **CLI Tools and CI/CD Integration**: For automation and deployment (e.g., using Jenkins or GitHub Actions).
2. **ETL Pipeline Stages**:
   * **Data Ingestion**:
     + Load CSV and XLSX files using PySpark.
   * **Data Cleaning**:
     + Clean and normalize data using PySpark transformations.
   * **Data Transformation**:
     + Use fuzzy matching to reconcile company names.
     + Generate unique IDs for companies where missing.
   * **Data Loading**:
     + Load the transformed data into PostgreSQL.
   * **Metadata Management**:
     + Track metadata for each data point.

### **2. Time Estimate**

1. **Requirement Analysis and Design (1 week)**:
   * Understand data sources, structures, and inconsistencies.
   * Design ETL pipeline architecture.
2. **Data Ingestion and Initial Cleaning (1 week)**:
   * Load CSV and XLSX files using PySpark.
   * Initial data cleaning and normalization.
3. **Data Transformation and Integration (2 weeks)**:
   * Implement fuzzy matching and merging logic.
   * Generate unique IDs and standardized data formats.
4. **Data Loading and Metadata Management (1 week)**:
   * Load transformed data to PostgreSQL.
   * Implement metadata tracking.
5. **Testing and Validation (1 week)**:
   * Develop unit tests for each ETL stage.
   * Perform data validation checks.
6. **CI/CD Integration and Documentation (1 week)**:
   * Integrate with CI/CD pipeline.
   * Document the ETL process and codebase.

**Total Estimated Time: 7 weeks**

### **3. Data Quality and Traceability**

* **Data Quality**:
  + Use PySpark for data cleaning and validation.
  + Regularly monitor data quality metrics.
* **Data Traceability**:
  + Maintain detailed metadata in PostgreSQL.
  + Track source file, load timestamp, and transformation history.
  + Implement logging in the ETL process.
* **Error Correction**:
  + Implement automated error detection and reporting.
  + Allow manual intervention for detected errors.

### **4. Data Integration**

* **Data Ingestion**:
  + Load CSV and XLSX files using PySpark.
* **Data Cleaning and Standardization**:
  + Clean and normalize data using PySpark.
* **Data Transformation**:
  + Use fuzzy matching to reconcile company names.
  + Generate unique IDs for companies and assets.
* **Unified Schema**:
  + Define a unified schema in PostgreSQL.