**Open-Ended Question**

**Parsing Data from Multiple XML Files to a Database**

**Context**

There are 12 faculties at a university. Every day, a new XML file is generated for each faculty. Each XML file contains structured data for all departments in that faculty. Within each department, the XML includes:

1. **Biodata of students (mahasiswa)**: Contains details such as name, student ID, program of study, etc.
2. **Biodata of lecturers (dosen)**: Contains details such as name, lecturer ID, department, specialization, etc.

The goal is to parse this data safely and efficiently into a database.

**Problem Statement**

You are tasked with designing a robust system to parse data from these daily XML files and insert the extracted data into a database. The system must ensure:

* **Data Integrity**: Prevent duplication or corruption of data during parsing and insertion.
* **Scalability**: Handle multiple XML files per day, as the number of faculties and departments grows.
* **Error Handling**: Address issues such as malformed XML files, missing fields, or unexpected data types.
* **Concurrency**: Safely handle scenarios where multiple XML files are processed simultaneously.
* **Security**: Protect the system from potential threats like XML External Entity (XXE) attacks or SQL injection during the database insertion process.

**Questions to Consider**

1. **Parsing Strategy**:
   * What tools or libraries would you use for parsing XML files? For example, xml.etree.ElementTree in Python, DOM parsers in Java, etc.
   * How would you handle deeply nested XML structures like those in this scenario (faculty → department → biodata)?
2. **Database Design**:
   * What schema would you propose for the database to store data for faculties, departments, students, and lecturers?
   * How would you enforce relationships between entities (e.g., faculty → department, department → students and lecturers)?
3. **Ensuring Data Integrity**:
   * How would you handle duplicate entries if an XML file is accidentally reprocessed?
   * How can you validate the data before inserting it into the database (e.g., schema validation, data type checks)?
4. **Handling Errors**:
   * How would your system handle malformed or incomplete XML files?
   * How would you log or report errors during parsing or database insertion?
5. **Concurrency and Scalability**:
   * How would you design the system to safely process multiple XML files simultaneously?
   * If the number of XML files grows over time, how can your system remain performant?
6. **Security Considerations**:
   * How would you protect the XML parser from XXE (XML External Entity) attacks?
   * How would you sanitize the data to prevent SQL injection or other database vulnerabilities?

**Deliverables**

* A **high-level architecture** or **workflow diagram** showing how XML files are parsed and data is inserted into the database.
* Sample **code snippets** or pseudo-code to demonstrate critical parts of your solution (e.g., XML parsing, database insertion, error handling).
* A **list of tools or libraries** you would use and why.
* A brief explanation of **how you would test the system** to ensure it meets the requirements above.

**Example Output**

Participants should provide a detailed and creative solution. For example:

* A Python script using xml.etree.ElementTree for parsing XML and SQLAlchemy for database interactions.
* A workflow diagram illustrating steps like XML validation, parsing, data transformation, and insertion.
* Suggested database schema and SQL queries.

This question evaluates the ability to design scalable, secure, and robust data processing systems.