Project Documentation: Zoo Management System

1.Introduction

The **Zoo Management System** project is designed to handle information related to zoos, animals, and their conservation status. By leveraging various XML-based technologies, the system enables efficient storage, transformation, and querying of data. The application combines a well-formed XML file with validation schemas, XSLT transformations, XQuery queries, a REST API, and a SOAP API to ensure comprehensive management and presentation of the data.

2. Project Structure

2.1 Main XML File (zoo.xml)

The XML file contains information about:

- **Zoos:** includes attributes such as id, name, city, foundation, and location.
- **Animals:** stores data such as id, species, zooid, name, scientific_name, habitat, and diet.
- **Conservation Statistics:** related to animals, includes details on population_in_wild, population_in_capacity and status.
- **Messages:** news or announcements related to zoos or animals.

The hierarchical structure ensures clear and accessible organization of the data.

3. XML Validation

The zoo.xml file is validated using the XML schema defined in zoo.xsd. This ensures that the data adheres to the specified structural and semantic rules, such as correct associations between zoos and animals.

The validation process is automated using the validate_xml.py script, which employs Python's lxml library to verify conformity with the schema.

4. XSLT Transformations

The system uses XSLT to transform data into more readable or visually appealing formats. The transformations include:

- **zoos_to_html.xsl**: generates an HTML table with information about zoos and their animals.
- **herbivores_to_html.xsl**: Creates a list of herbivorous animals.
- **animals_by_habitat.xsl**: Groups animals by habitat.
- **endangered_animals.xsl**: Displays a report of endangered animals.
- **animals_count_by_zoo.xsl**: Shows a count of animals per zoo.

The transformations.py script automates these transformations, generating the corresponding HTML files.

5. eXistDB Queries

For advanced queries, eXistDB is used in conjunction with existdb_queries.py, which defines the following Xquery queries:

1. List of herbivorous animals
2. Information about zoos
3. Animals at risk of extinction
4. Count of animals per zoo
5. Animals in specific habitats.

These queries enable quick and efficient retrieval of relevant information.

6. API Interface

6.1. api.py

This is the REST API that is used to integrate with Xforms to invoke different operations that utilize said API.

- Retrieve animals filtered by zoo and species using query parameters.

6.2. app.py

The Flask application defines CRUD routes for zoos, animals, and conservation statistics. These functionalities are backed by functions in xml_utils.py for XML file manipulation.

7.XML Utilities

The xml_utils.py file includes key functions such as:

- Zoos: retrieve, add, update and delete zoos
- Animals: manage animal information
- Statistics: handle conservation statistics

These utilities ensure efficient and secure data manipulation.

8.Web Interface

The interface defined in index.html allows to interact with the system intuitively. Some functionalities include:

- Listing zoos and animals
- Adding new entries
- Displaying detailed information

9. Generated Outputs

The system produces results in HTML and XML formats:

- Files generated through XSLT transformations
- API REST responses
- Outputs from Xquery queries executed in eXistDB.