Reg.No:22IT068

Practice with XML DTD and XSD

Ex No: 8b Date: 19.09.24

# **QUESTION**

- 1. Create an XML file to store details of students (name, age, course, grade). Ensure that the file has at least 3 entries.
  - a. How can you modify the structure to include optional elements like address or phone number?
- 2. Write a DTD for an XML file that defines a bookstore.
  - a. The bookstore should contain elements for <book>, with child elements like <title>, <author>, <year>, and <price>.
  - b. Ensure that every book has a title and author, but price is optional.
- 3. Create an XML document representing a company's employee database. Use DTD to ensure that each employee element contains a name, ID, department, and an optional address.
  - a. What happens if you try to add an invalid or missing element in the XML file?
- 4. Write a DTD that validates an XML document for a recipe book. Each recipe should contain a title, an ingredient list, and instructions. Ingredients can have optional attributes like quantity and unit.
  - a. How can you enforce that each recipe contains at least one ingredient and one instruction?
- 5. Using internal DTD, create an XML document that represents a catalog of movies, with elements for title, director, genre, and release year.
  - a. How would you enforce that the release year is a four-digit number using DTD?
- 6. Design an XML and corresponding DTD to represent a product inventory system. Each product must have a unique identifier, name, category, and price. Optionally, it may have a description and stock quantity.
  - a. How can you make the unique identifier a required attribute using DTD?
- 7. Modify an XML document to use an external DTD instead of an internal one.
  - a. What changes are necessary in the XML file to reference the external DTD?

## **AIM**

To practice XML DTD along with schema.

## **CODE**

```
1.<students>
<student>
<name>John Doe</name>
<age>20</age>
```

<course>Information Technology</course>

```
Reg.No:22IT068
```

```
<grade>A</grade>
    <address>123 Main Street</address>
    <phone>123-456-7890</phone>
  </student>
  <student>
    <name>Jane Smith</name>
    <age>22</age>
    <course>Computer Science</course>
    <grade>B+</grade>
    <!-- Optional elements can be omitted -->
  </student>
  <student>
    <name>Mark Johnson</name>
    <age>21</age>
    <course>Mechanical Engineering</course>
    <grade>A-</grade>
    <address>456 Park Avenue</address>
  </student>
</students>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="students">
    <xs:complexType>
      <xs:sequence>
         <xs:element name="student" maxOccurs="unbounded">
           <xs:complexType>
             <xs:sequence>
               <xs:element name="name" type="xs:string" />
               <xs:element name="age" type="xs:integer" />
               <xs:element name="course" type="xs:string" />
               <xs:element name="grade" type="xs:string" />
               <xs:element name="address" type="xs:string" minOccurs="0" />
               <xs:element name="phone" type="xs:string" minOccurs="0" />
```

```
Reg.No:22IT068
                 </xs:sequence>
              </r></rs:complexType>
           </xs:element>
        </xs:sequence>
      </r></rs:complexType>
  </xs:element>
</xs:schema>
OUTPUT
     The XML document is valid.
     Option 1: Copy-paste your XML document here
      <students>
        <student>
          <name>John Doe</name>
          <age>20</age>
          <course>Information Technologys/course>
     Option 2: Or upload your XML file
                                                                   File encoding
       Choose File
                    No file chosen
                                                                     UTF-8
     Option 1: Copy-paste your XSD here (Optional if XSD referred in XML using schemaLocation)
      <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
        <xs:element name="students">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="student" maxOccurs="unbounded">
     Option 2: Or upload your XSD document
                                                                    File encoding
                                                                     UTF-8
       Choose File
                    No file chosen
       Validate XML
```

2. <?xml version="1.0"?>

<!DOCTYPE bookstore [</pre>

<!ELEMENT bookstore (book+)>

<!ELEMENT title (#PCDATA)>

<!ELEMENT author (#PCDATA)>

<!ELEMENT book (title, author, year, price?)>

```
<!ELEMENT year (#PCDATA)>
  <!ELEMENT price (#PCDATA)>
]>
<br/>bookstore>
  <book>
    <title>Introduction to Algorithms</title>
    <author>Thomas H. Cormen</author>
    <year>2009</year>
    <price>89.99</price>
  </book>
  <book>
    <title>The Pragmatic Programmer</title>
    <author>Andrew Hunt</author>
    <year>1999</year>
    <!-- Price is omitted, as it is optional -->
  </book>
  <book>
    <title>Clean Code</title>
    <author>Robert C. Martin</author>
    <year>2008</year>
    <price>44.99</price>
  </book>
</bookstore>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="bookstore">
    <xs:complexType>
      <xs:sequence>
         <xs:element name="book" maxOccurs="unbounded">
           <xs:complexType>
             <xs:sequence>
                <xs:element name="title" type="xs:string" />
                <xs:element name="author" type="xs:string" />
```

```
Reg.No:22IT068
                    <xs:element name="year" type="xs:integer" />
                    <xs:element name="price" type="xs:decimal" minOccurs="0" />
                 </xs:sequence>
              </r></rs:complexType>
           </xs:element>
        </xs:sequence>
     </r></rs:complexType>
  </xs:element>
</xs:schema>
OUTPUT
     The XML document is valid.
     Option 1: Copy-paste your XML document here
      <?xml version="1.0"?>
      <!DOCTYPE bookstore [</pre>
        <!ELEMENT bookstore (book+)>
        <!ELEMENT book (title, author, year, price?)>
        <IFLEMENT title (#PCDATA)>
     Option 2: Or upload your XML file
                                                                  File encoding
      Choose File
                   No file chosen
                                                                   UTF-8
     Option 1: Copy-paste your XSD here (Optional if XSD referred in XML using schemaLocation)
      <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
        <xs:element name="bookstore">
```

# 3. <?xml version="1.0"?> <!DOCTYPE company [ <!ELEMENT company (employee+)> <!ELEMENT employee (name, id, department, address?)>

File encoding

UTF-8

<xs:complexType>
<xs:sequence>

Choose File

Validate XML

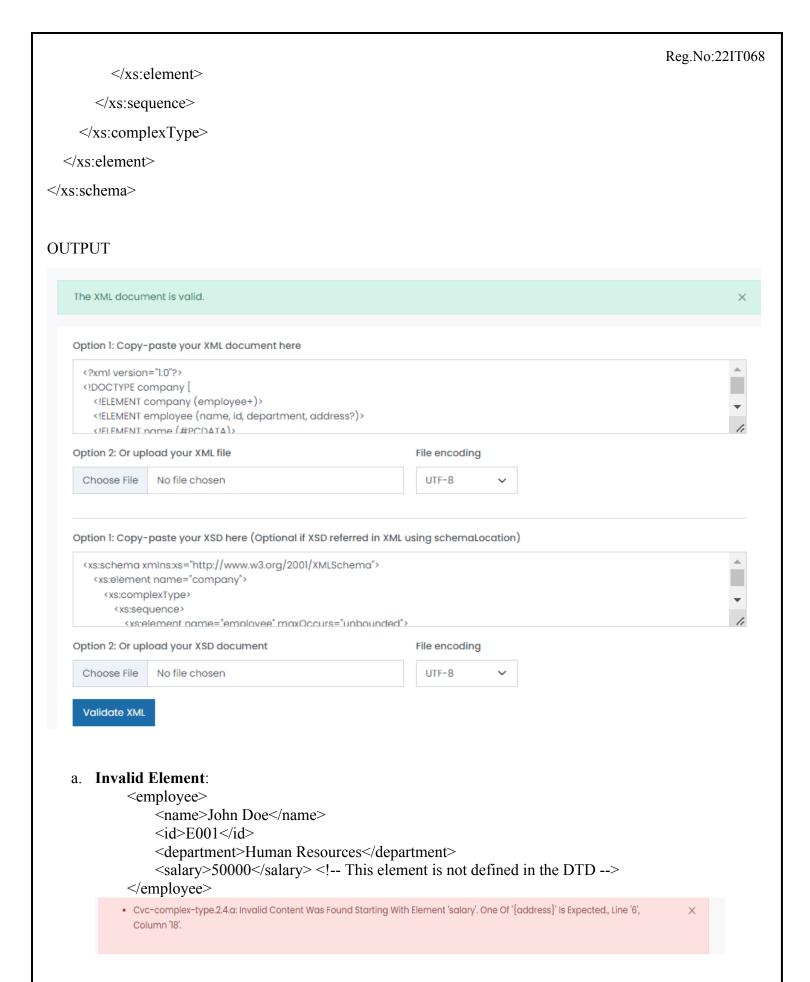
Option 2: Or upload your XSD document

No file chosen

<xs:element name="book" maxOccurs="unbounded">

Reg.No:22IT068

```
<!ELEMENT name (#PCDATA)>
  <!ELEMENT id (#PCDATA)>
  <!ELEMENT department (#PCDATA)>
  <!ELEMENT address (#PCDATA)>
]>
<company>
  <employee>
    <name>John Doe</name>
    <id>E001</id>
    <department>Human Resources</department>
    <address>123 Main Street</address>
  </employee>
  <employee>
    <name>Jane Smith</name>
    <id>E002</id>
    <department>Engineering</department>
    <!-- Address is optional -->
  </employee>
</company>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="company">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="employee" maxOccurs="unbounded">
           <xs:complexType>
             <xs:sequence>
               <xs:element name="name" type="xs:string" />
               <xs:element name="id" type="xs:string" />
               <xs:element name="department" type="xs:string" />
               <xs:element name="address" type="xs:string" minOccurs="0" />
             </xs:sequence>
           </r></rs:complexType>
```



## **Missing Element:**

```
<employee>
     <name>Jane Smith</name>
     <!-- ID is missing -->
          <department>Engineering</department>
</employee>
```

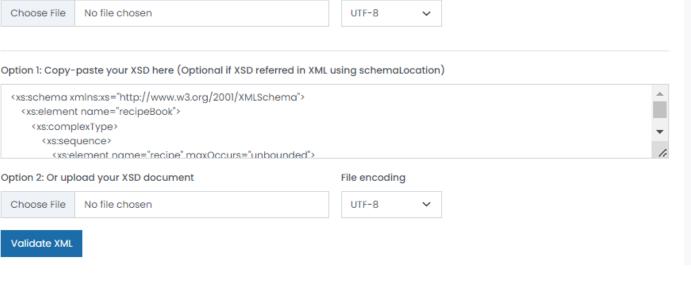
Cvc-complex-type.2.4.a: Invalid Content Was Found Starting With Element 'department'. One Of '{id}' Is Expected., Line '5',
 Column '21'.

 $\times$ 

```
4. <?xml version="1.0"?>
<!DOCTYPE recipeBook [</pre>
  <!ELEMENT recipeBook (recipe+)>
  <!ELEMENT recipe (title, ingredients, instructions)>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT ingredients (ingredient+)>
  <!ELEMENT ingredient (#PCDATA)>
  <!ATTLIST ingredient
    quantity CDATA #IMPLIED
    unit CDATA #IMPLIED>
  <!ELEMENT instructions (instruction+)>
  <!ELEMENT instruction (#PCDATA)>
]>
<recipeBook>
  <recipe>
    <title>Spaghetti Bolognese</title>
    <ingredients>
       <ingredient quantity="500" unit="g">Spaghetti</ingredient>
       <ingredient quantity="250" unit="g">Minced beef</ingredient>
       <ingredient>Tomato sauce</ingredient>
    </ingredients>
    <instructions>
       <instruction>Boil the spaghetti.</instruction>
       <instruction>Cook the minced beef.</instruction>
```

```
<instruction>Mix with tomato sauce and serve./instruction>
    </instructions>
  </recipe>
</recipeBook>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="recipeBook">
    <xs:complexType>
      <xs:sequence>
         <xs:element name="recipe" maxOccurs="unbounded">
           <xs:complexType>
             <xs:sequence>
                <xs:element name="title" type="xs:string"/>
                <xs:element name="ingredients">
                  <xs:complexType>
                    <xs:sequence>
                       <xs:element name="ingredient" maxOccurs="unbounded">
                         <xs:complexType>
                           <xs:simpleContent>
                              <xs:extension base="xs:string">
                                <xs:attribute name="quantity" type="xs:string" use="optional"/>
                                <xs:attribute name="unit" type="xs:string" use="optional"/>
                              </xs:extension>
                           </xs:simpleContent>
                         </r></rs:complexType>
                       </xs:element>
                    </xs:sequence>
                  </xs:complexType>
                </xs:element>
                <xs:element name="instructions">
                  <xs:complexType>
                    <xs:sequence>
                       <xs:element name="instruction" maxOccurs="unbounded" type="xs:string"/>
```





a. By declaring ingredient+ and instruction+, the DTD enforces that there must be at least one ingredient

maxOccurs="unbounded" in the ingredient and instruction elements.

The structure enforces that each recipe contains at least one ingredient and one instruction through the

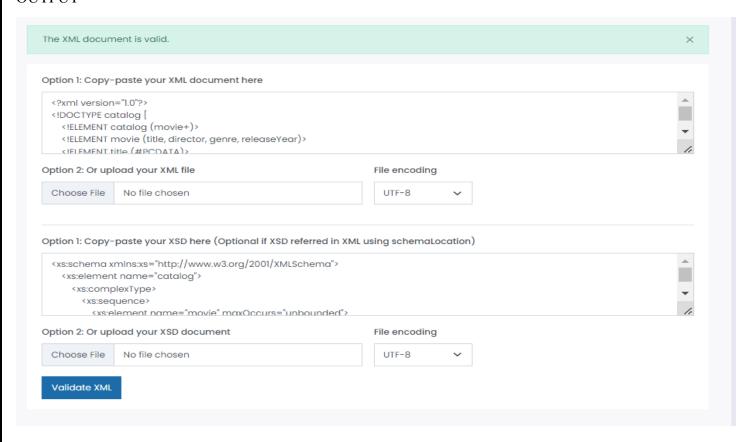
and one instruction in each recipe.

```
5. <?xml version="1.0"?>
<!DOCTYPE catalog [
  <!ELEMENT catalog (movie+)>
  <!ELEMENT movie (title, director, genre, releaseYear)>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT director (#PCDATA)>
  <!ELEMENT genre (#PCDATA)>
  <!ELEMENT releaseYear (#PCDATA)>
]>
<catalog>
  <movie>
    <title>Inception</title>
    <director>Christopher Nolan</director>
    <genre>Sci-Fi</genre>
    <releaseYear>2010</releaseYear>
  </movie>
  <movie>
    <title>The Matrix</title>
    <director>Lana Wachowski, Lilly Wachowski</director>
    <genre>Action</genre>
    <releaseYear>1999</releaseYear>
  </movie>
</catalog>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="catalog">
    <xs:complexType>
      <xs:sequence>
         <xs:element name="movie" maxOccurs="unbounded">
           <xs:complexType>
             <xs:sequence>
               <xs:element name="title" type="xs:string"/>
```

```
Reg.No:22IT068
```

```
<xs:element name="director" type="xs:string"/>
                <xs:element name="genre" type="xs:string"/>
                <xs:element name="releaseYear">
                  <xs:simpleType>
                    <xs:restriction base="xs:string">
                       <xs:pattern value="\d{4}"/>
                    </xs:restriction>
                  </xs:simpleType>
                </xs:element>
             </xs:sequence>
           </r></re>
         </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

## **OUTPUT**



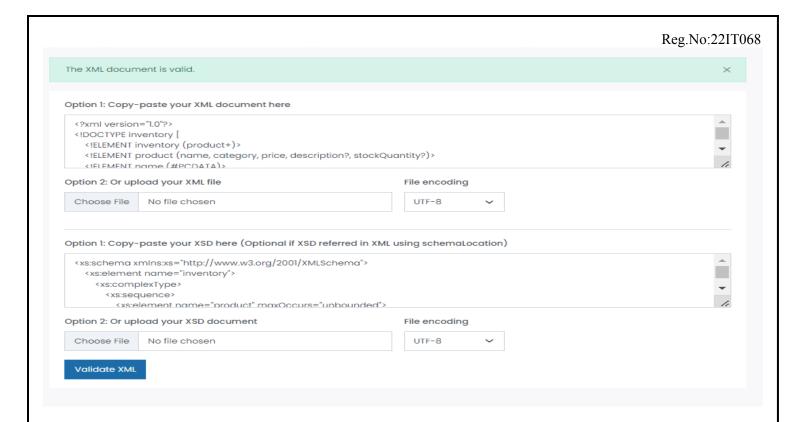
a. DTD ensures the structure is followed; it cannot enforce that releaseYear is a four-digit number. To enforce such a restriction, you would need to use an XML Schema (XSD) xs:pattern value="\d{4}": Ensures that releaseYear contains exactly four digits.

```
6. <?xml version="1.0"?>
<!DOCTYPE inventory [
  <!ELEMENT inventory (product+)>
  <!ELEMENT product (name, category, price, description?, stockQuantity?)>
  <!ELEMENT name (#PCDATA)>
  <!ELEMENT category (#PCDATA)>
  <!ELEMENT price (#PCDATA)>
  <!ELEMENT description (#PCDATA)>
  <!ELEMENT stockQuantity (#PCDATA)>
  <!ATTLIST product id ID #REQUIRED>
]>
<inventory>
  cproduct id="P001">
    <name>Smartphone</name>
    <category>Electronics</category>
    <price>699.99</price>
    <description>High-end smartphone with 128GB storage</description>
    <stockQuantity>50</stockQuantity>
  </product>
  cproduct id="P002">
    <name>Laptop</name>
    <category>Computers</category>
    <price>999.99</price>
    <!-- Description and stock quantity are optional -->
  </product>
</inventory>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

```
Reg.No:22IT068
```

```
<xs:element name="inventory">
    <xs:complexType>
      <xs:sequence>
         <xs:element name="product" maxOccurs="unbounded">
           <xs:complexType>
             <xs:sequence>
               <xs:element name="name" type="xs:string"/>
               <xs:element name="category" type="xs:string"/>
               <xs:element name="price" type="xs:decimal"/>
               <xs:element name="description" type="xs:string" minOccurs="0"/>
               <xs:element name="stockQuantity" type="xs:integer" minOccurs="0"/>
             </xs:sequence>
             <xs:attribute name="id" type="xs:ID" use="required"/>
           </r></re>
         </xs:element>
      </xs:sequence>
    </r></rs:complexType>
  </xs:element>
</xs:schema>
```

**OUTPUT** 



a. By declaring the id attribute as type ID, you enforce that it is unique and required for each product xs:ID for the id attribute ensures it is a unique identifier and is required.

7.

- ➤ Using External DTD: Move the DTD declarations to a .dtd file and reference it in the DOCTYPE declaration in the XML file.
  - <!DOCTYPE parentclass SYSTEM "filename.dtd">
- > **XSD**: The XSD defines the same structure but with better data type control, ensuring that attributes and element types are validated.
- In this case, the external DTD is referenced via the SYSTEM keyword, while the XSD is referenced using the xsi:noNamespaceSchemaLocation attribute in the XML.

	Reg.No:22IT068
	C
RESULT	
Thus, practiced and validated XML,DTD along with schema(XSD).	