TRANSFORM.PY

from lxml import etree

This line imports the etree module from the lxml library. The etree module provides functionalities for processing XML and performing transformations.

xml\_tree = etree.parse("amazon.xml")

The etree.parse() function is used to parse the XML file "amazon.xml" and create an ElementTree object named xml\_tree. This represents the hierarchical structure of the XML document.

xsl\_transform = etree.XSLT(etree.parse("amazon.xsl"))

The XSLT stylesheet, "amazon.xsl," is parsed using etree.parse(), and an XSLT object named xsl\_transform is created. This object represents the compiled XSLT stylesheet and can be used to apply transformations.

html\_tree = xsl\_transform(xml\_tree)

The xsl\_transform object is applied to the xml\_tree representing the XML document. The result of the transformation is an ElementTree object named html\_tree, which contains the transformed HTML content.

with open("amazon.html", "wb") as output\_file:

    output\_file.write(etree.tostring(html\_tree, pretty\_print=True))

The transformed HTML content stored in the html\_tree object is converted to a byte string using etree.tostring(). The resulting HTML content is then written to a file named "amazon.html" in binary write mode ("wb"). The pretty\_print=True argument ensures that the generated HTML is formatted with proper indentation for readability.

VALIDATION.PY

import xmlschema

This line imports the xmlschema module, which is a Python library used for working with XML schemas and validating XML documents against them.

xml\_file = "amazon.xml"

xsd\_file = "amazon.xsd"

Here, xml\_file and xsd\_file are strings that store the paths to the XML file and the XSD file, respectively.

if validator.is\_valid(xml\_file):

    print("XML File is valid against the XSD Schema.")

else:

    print("XML File is not valid against the XSD Schema.")

The is valid method of the validator instance is used to check whether the provided XML file adheres to the rules defined in the XSD schema. If the XML is valid, it prints a message indicating that the XML is valid against the XSD schema. If not, it prints a message indicating that the XML is not valid against the XSD schema.

try:

        print(validator.validate(xml\_file))

    except xmlschema.validators.exceptions.XMLSchemaDecodeError as exception:

        print(exception.reason)

If the XML file is not valid, the script attempts to validate it again using the validate method. If validation fails, a XMLSchemaDecodeError exception might be raised. This exception is caught, and its reason attribute is printed, providing more details about why the validation failed.

This script uses the xmlschema library to validate an XML file against an XSD schema. If the XML is valid, it outputs a success message. If it's not valid, it prints an error message and, if available, additional details about the validation error.

**TESTING**

Here I tried to add <quantity> which is not defined in my ShopSmart.xsd so it is showing as error.

<product>

        <id>3</id>

        <name>Laptop</name>

        <price>499.99</price>

        <description>New Laptop</description>

        <quantity>2</quantity>

    </product>

<xs:sequence>

                            <xs:element name="id" type="xs:int" />

                            <xs:element name="name" type="xs:string" />

                            <xs:element name="price" type="xs:float" />

                            <xs:element name="description" type="xs:string" />

                        </xs:sequence>

So when I try to validate it. It gives me an Error.

**Reason**: Unexpected child with tag 'quantity' at position 5.

**Schema**:

<xs:complexType xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:sequence>

<xs:element name="id" type="xs:int" />

<xs:element name="name" type="xs:string" />

<xs:element name="price" type="xs:float" />

<xs:element name="description" type="xs:string" />

</xs:sequence>

</xs:complexType>

Instance:

<product>

<id>3</id>

<name>Laptop</name>

<price>499.99</price>

<description>New Laptop</description>

<quantity>2</quantity>

</product>

**TRANSFORMATION ERROR**

Below I have provided the name as Integer instead of String and tried to use Transform.py.

<product>

        <id>1</id>

        <name>12345</name>

        <price>799.9</price>

        <description>New Mobile Phone</description>

    </product>

Transform.py didn’t throw any errors but when we save the file and try to run ShopSmart.html file

The output content is not accurately displayed.

