Module Code & Module Title

CS5004NI Emerging Programming Platforms and Technologies

Assessment Weightage & Type

30% Individual Coursework

Year and Semester

2019-20 Autumn / 2020-21 Spring

Student Name: Preksha Dahal

London Met ID: 20048964

College ID: np01cp4s210067

Group: C13

Assignment Due Date: 5th May 2022

Assignment Submission Date: 5th May 2022

Title (Where Required): Total (XML)

Word Count (Where Required): 2000

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Table of Contents

Table of Figures

List of Tables

Introduction

Extensible markup language or XML is a markup language similar to Hypertext Markup Language (HTML). A markup language is basically used to specify the structure and content of a particular document. Since XML is an extensible markup language it can be used to create a wide variety of document types. The extensible markup language can also be defined as the subset of Standard Generalized Markup Language (SGML) which was introduced in the year 1986. This markup language was extensible to Hypertext Markup Language (HTML) and, Extensible Markup Language (XML). SGML was very costly and complex due to which new markup languages like HTML, DTD, GML, and many others. XML can be placed in between SGML and HTML as it is easier to learn than standard generalized markup language but tougher than hypertext markup language. Since extensible markup language is an extensible markup language it helps to solve the data description and cataloging issues. It can be applied consistently, and the elements and attributes can be modified easily. The main goals of an XML design are;

- It should be easily usable over the Internet.
- It should support a wide variety of applications.
- It should be compatible with SGML, CSS, and XSD.
- It should be easy to write and execute.
- The number of optional features should not exceed a large value.
- The document should be easily and clearly understood by the non-programmers.
- The design should be exact and concise.

XML includes various subsets like Channel Definition Format (CDF), Chemical Markup Language (CML), Extensible Hypertext Markup Language (XHTML), Mathematical Markup Language (MathML), Musical Markup Language (MML), Voice Markup Language (VXML), etc.

The XML document displays only a set of codes in a browser. To format the document, we need to link the document to a style sheet. After linking the XML document with a style sheet, the XML processor will combine the style sheet with the XML document and apply all the formatting codes defined in the style sheet to display the formatted document. Cascading Style Sheet (CSS) and Extensible Style Sheet are the two main style sheet languages used with XML. The correct way of linking an XML file with the style sheet is given below;

```
<!--Linking the style sheet with XML-->
<?xml-stylesheet type="text/css" href="catalog_20048964.css"?>
```

Figure 1: Linking CSS to XML

Cascading Style Sheet (CSS) is a language that is used to style a markup language like HTML and XML. The style sheet basically defines the formatting of the XML document.

Like CSS (Cascading Style Sheet), another language (or file) known as XML schema can also be linked to XML files. They are known as XML Schema. XML schema uses the extension XSD which stands for extensible schema definition. An XML schema is used to describe the structure of an XML document. It acts as a blueprint of the XML document.

```
<!--Linking the schema with XML-->
<?xml-model href="catalog_20048964.xsd" type="application/xml" schematypens="http://www.w3.org/2001/XMLSchema"?>
```

Figure 2: Linking Schema to XML document.

XML Document

List of Data

The data of XML document includes the details of its elements and the attributes.

List of Elements

The description of the elements are;

S.N.	List of Element	Element Type	Element Data	Occurrence
1.	**giftCards**	Complex type	Child element	Only once
2.	**store**	Complex type	Child element	Only once
3.	**nav**	Complex type	Child element	Only once
4.	**logo**	Complex type	Child element	Only once
5.	**option1**	Simple type	String	Only once

6.	**option2**	Simple type	String	Only once
7.	**Option3**	Simple type	String	Only once
8.	**searchbar**	Complex type	Child element	Only once
9.	**details**	Complex type	Child element	Only Once
10.	**Slogan**	Complex Type	String	Only Once
11.	**cards**	Complex Type	Child element	Only Once
12.	**bestSeller**	Complex Type	Child element	Only once
13.	**heading**	Simple Type	String	One or more
14.	**items**	Complex Type	Child element	One or more
15.	**card**	Complex Type	Child element	One or more
16.	**cardName**	Simple Type	String	One or more

17.	**type**	Simple Type	String	One or more
18.	**cost**	Complex Type	Child element	One or more
19.	**availableNumber**	Simple Type	positiveInteger	Zero or one
20.	**no_of_users**	Simple Type	positiveInteger	One or more
21.	**discount**	Complex Type	Child element	Zero or one
22.	**discount_vouchers**	Simple Type	String	One or more
23.	**refund**	Complex Type	Child element	Zero or one
24.	**Refund**	Simple Type	String	One or more
25.	**company**	Complex Type	Child element	Zero or one
26.	**Company**	Simple Type	String	One or more
27.	**otherCards**	Complex Type	Child element	Only once

28.	**footer**	Complex Type	Child element	Only once
29.	**storeName**	Simple Type	String	Only once
30.	**address**	Complex Type	Child element	Only once
31.	**text**	Simple Type	String	One or more
32.	**telephone**	Complex Type	Child element	Only once
33.	**msg1**	Simple type	String	Only once
34.	**follow**	Complex Type	Child element	Only once
35.	**msg**	Simple Type	String	One or more
36.	**vat**	Simple type	String	One or more
37.	**image**	Complex type	Empty element	Only once
38.	**name**	Simple type	String	Only once

39.	**URL**	Simple type	String	Only once
40.	**img**	Complex type	Empty element	Only once
41.	**validity**	Simple type	gYear	Only once
42.	**renew**	Complex type	Child element	Zero or one
43.	**Renew**	Simple type	String	Zero or one
44.	**logoText**	Simple type	String	Only once

Table 1: List of Elements

List of Attributes

The description of the attributes are;

S.N.	List of Attribute	Datatype	Default	Fixed	Use
1.	**cardId**	ID	-	-	Required
2.	**id**	String	-	-	Required
3.	**option**	String	-	-	Required
4.	**Available**	Boolean	true	-	Optional
5.	**choice**	String	-	-	Required
6.	**name**	String	-	-	Required
7.	**currency**	String	-	$	Optional

Table 2: List of Attributes

Tree Diagram

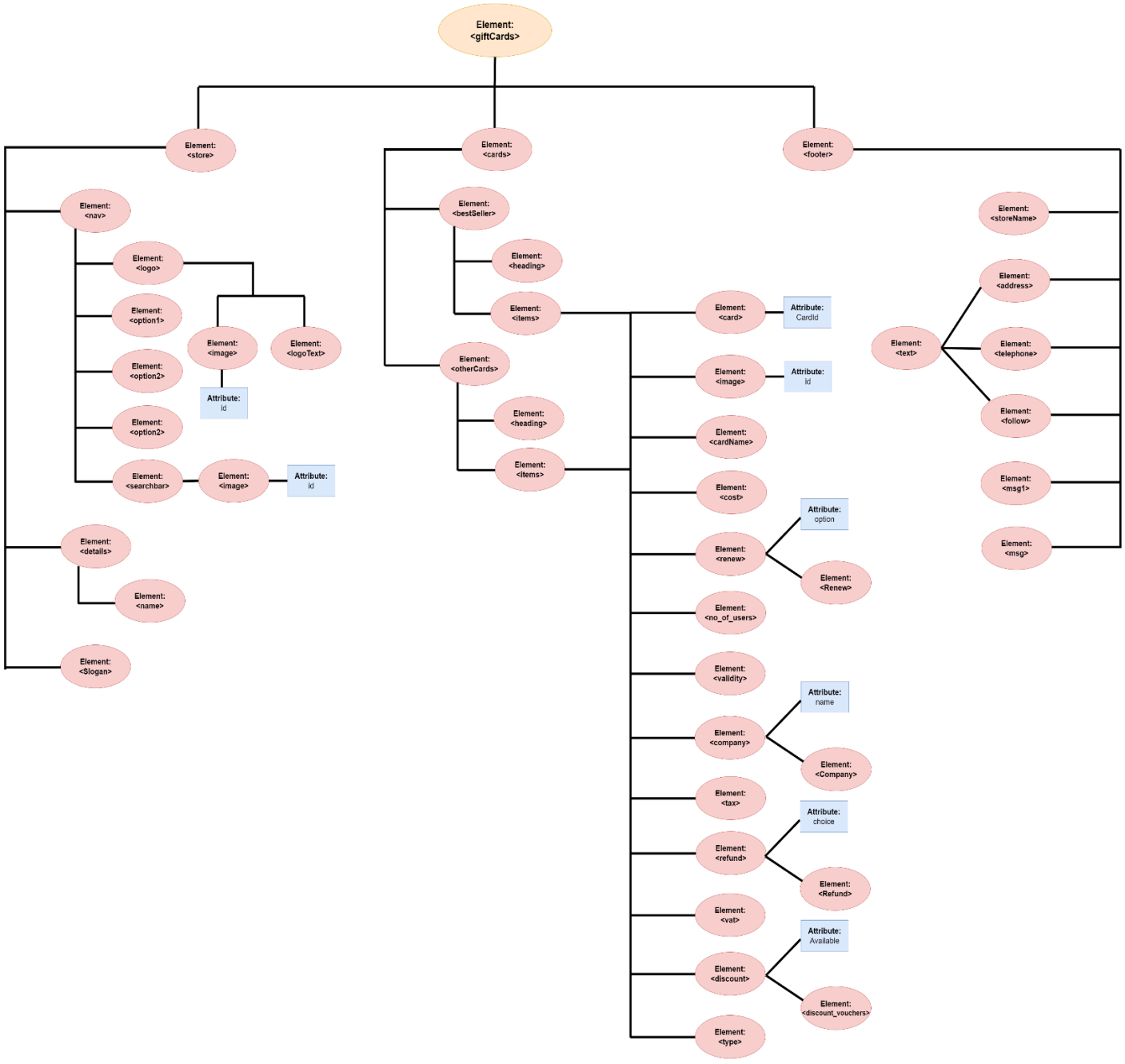

Figure 3: Tree Diagram of the XML document

XML Code

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<!--Linking the style sheet with XML-->
<?xml-stylesheet type="text/css" href="catalog_20048964.css"?>

<!--Linking the XML schema with XML-->
<?xml-model href="catalog_20048964.xsd" type="application/xml"
schematypens="http://www.w3.org/2001/XMLSchema"?>

<giftCards>
    <store>
        <nav>
            <logo>
                <image id="aesthecticLogo" />
                <logoText>Aesthectic.ly~~</logoText>
            </logo>
            <option1>Home</option1>
            <option2>About Us</option2>
            <option3>Cards</option3>
            <searchbar>
                <image id="searchbar" />
            </searchbar>
        </nav>
        <details>
            <name>Aesthectic.ly Cards </name>
        </details>
        <Slogan>
                We believe in doing it Aesthectical.ly!!
        </Slogan>
        <URL>[ https://aesthetic.lly.com/ ]</URL>
    </store>


    <cards>
        <bestSeller>
            <heading>Some of our Best-sellers cards are;</heading>
            <items>
                <card cardId="card01">
                    <img id="card1" />
                    <cardName>Dell</cardName>
```

```
        <type>Physical</type>
        <cost>400</cost>
        <validity>2025</validity>
        <availableNumber>12</availableNumber>
        <no_of_users>15</no_of_users>
        <discount Available="false"></discount>
        <refund choice="No">
            <Refund>No</Refund>
        </refund>
        <company name="Dell">
            <Company>DELL</Company>
        </company>
    </card>


    <card cardId="card02">
        <!--card01 is for card details-->
        <img id="card2" />
        <!--card1 is for image-->
        <cardName>Preety Click</cardName>
        <type>Physical</type>
        <cost>2000</cost>
        <validity>2026</validity>
        <availableNumber>2</availableNumber>
        <no_of_users>115</no_of_users>
        <renew option="Yes">
            <Renew>Will be available soon</Renew>
        </renew>
        <discount>
            <discount_vouchers>10%</discount_vouchers>
        </discount>
        <refund choice="Yes">
            <Refund>Yes</Refund>
        </refund>
    </card>

    <card cardId="card_03">
        <img id="card3" />
        <cardName>The Assets</cardName>
        <type>Both</type>
        <cost>4000</cost>
        <validity>2025</validity>
        <no_of_users>12</no_of_users>
```

```
                <renew option="No">
                    <Renew>No</Renew>
                </renew>
                <discount>
                    <discount_vouchers>12%</discount_vouchers>
                </discount>
                <vat>12%</vat>
                <refund choice="No">
                    <Refund>No</Refund>
                </refund>
                <company name="The Assets">
                    <Company>The Assets</Company>
                </company>
            </card>
        </items>
    </bestSeller>

    <otherCards>
        <heading>Other Samples</heading>
        <items>
            <card cardId="card04">
                <img id="card4" />
                <cardName>Google Play </cardName>
                <type>Digital</type>
                <cost>25</cost>
                <validity>2023</validity>
                <availableNumber>25</availableNumber>
                <no_of_users>17</no_of_users>

                <renew option="No">
                    <Renew>No</Renew>
                </renew>
                <discount>
                    <discount_vouchers>13%</discount_vouchers>
                </discount>
            </card>


            <card cardId="card05">
                <img id="card5" />
                <cardName>ebay</cardName>
                <type>Physical</type>
                <cost>27</cost>
```

```
                <validity>2024</validity>
                <availableNumber>2</availableNumber>
                <no_of_users>18</no_of_users>
                <discount Available="false"></discount>
                <refund choice="Yes">
                    <Refund>Yes</Refund>
                </refund>
                <company name="EBAY">
                    <Company>EBAY</Company>
                </company>
            </card>

            <card cardId="card06">
                <img id="card6" />
                <cardName>iTunes</cardName>
                <type>Digital</type>
                <cost>25</cost>
                <validity>2027</validity>
                <availableNumber>20</availableNumber>
                <no_of_users>8</no_of_users>
                <renew option="Yes">
                    <Renew>Yes</Renew>
                </renew>
                <discount>
                    <discount_vouchers>13%</discount_vouchers>
                </discount>
                <vat>12%</vat>
            </card>
        </items>

        <items>
            <card cardId="card07">
                <img id="card7" />
                <cardName>ebay Updated</cardName>
                <type>Both</type>
                <cost>123</cost>
                <validity>2042</validity>
                <no_of_users>45</no_of_users>
                <renew option="Yes">
                    <Renew>Yes</Renew>
                </renew>
                <discount>
```

```
                    <discount_vouchers>18%</discount_vouchers>
                </discount>
                <refund choice="Yes">
                    <Refund>Yes</Refund>
                </refund>
            </card>

            <card cardId="card08">
                <img id="card8" />
                <cardName>Roblox</cardName>
                <type>Digital</type>
                <cost>10</cost>
                <validity>2023</validity>
                <availableNumber>70</availableNumber>
                <no_of_users>78</no_of_users>
                <renew option="Will be available soon">
                    <Renew>Will be available soon</Renew>
                </renew>
                <discount Available="false"></discount>
            </card>

            <card cardId="card09">
                <img id="card9" />
                <cardName>Amazon Gifts</cardName>
                <type>Digital</type>
                <cost>150</cost>
                <validity>2040</validity>
                <no_of_users>52</no_of_users>
                <discount>
                    <discount_vouchers>10%</discount_vouchers>
                </discount>
                <company name="Amazon">
                    <Company>Amazon</Company>
                </company>
            </card>
        </items>

        <items>
            <card cardId="card010">
                <img id="card10" />
                <cardName>Windows</cardName>
                <type>Physical</type>
```

```
        <cost>50</cost>
        <validity>2026</validity>
        <no_of_users>45</no_of_users>
        <renew option="No">
            <Renew>No</Renew>
        </renew>
        <discount Available="false"></discount>
        <vat>12%</vat>
    </card>

    <card cardId="card011">
        <img id="card11" />
        <cardName>Chegg</cardName>
        <type>Both</type>
        <cost>40</cost>
        <validity>2025</validity>
        <availableNumber>250</availableNumber>
        <no_of_users>15</no_of_users>
        <renew option="Yes">
            <Renew>Yes</Renew>
        </renew>
        <discount>
            <discount_vouchers>15%</discount_vouchers>
        </discount>
        <vat>12%</vat>
    </card>

    <card cardId="card012">
        <img id="card12" />
        <cardName>Xbox</cardName>
        <type>Physical</type>
        <cost>75</cost>
        <validity>2028</validity>
        <no_of_users>30</no_of_users>
        <discount>
            <discount_vouchers>5%</discount_vouchers>
        </discount>
        <refund choice="No">
            <Refund>No</Refund>
        </refund>
        <company name="XBOX">
            <Company>XBOX</Company>
```

```
            </company>
        </card>
    </items>

    <items>
        <card cardId="card013">
            <img id="card13" />
            <cardName>Netfix Combined😍 </cardName>
            <type>Both</type>
            <cost>200</cost>
            <validity>2040</validity>
            <availableNumber>80</availableNumber>
            <no_of_users>180</no_of_users>
            <renew option="Will be available soon">
                <Renew>Will be available soon</Renew>
            </renew>
            <discount Available="false"></discount>
            <refund choice="Yes">
                <Refund>Yes</Refund>
            </refund>
        </card>

        <card cardId="card014">
            <img id="card14" />
            <cardName>iTunes Super Mega</cardName>
            <type>Both</type>
            <cost>210</cost>
            <validity>2045</validity>
            <no_of_users>120</no_of_users>
            <discount Available="false"></discount>
            <refund choice="No">
                <Refund>No</Refund>
            </refund>
            <company name="Apple">
                <Company>Apple</Company>
            </company>
        </card>

        <card cardId="card015">
            <img id="card15" />
            <cardName>iTunes Mega</cardName>
            <type>Digital</type>
```

```
                    <cost>170</cost>
                    <validity>2041</validity>
                    <no_of_users>25</no_of_users>
                    <renew option="No">
                        <Renew>No</Renew>
                    </renew>
                    <discount>
                        <discount_vouchers>3%</discount_vouchers>
                    </discount>
                    <vat>12%</vat>
                </card>
            </items>
        </otherCards>
    </cards>

    <footer>
        <storeName>Aesthetic.lly</storeName>
        <address>
            <text>Kathmandu | Pokhara | Biratnagar | Jhapa |
Birgunj</text>
        </address>
        <telephone>
            <text>9800000001 | 981200002 | 912500000 | 984532125 |
9865432234</text>
        </telephone>
        <msg1>Please note that the contact numbers are placed sequentially
with our location.</msg1>
        <follow>
            <text>Instagram | Facebook | Messenger | Viber | Tiktok</text>
        </follow>
        <msg>
            Offering several gift cards since 2001.
        </msg>
        <msg>
            Copyright © 2022 Aesthetic.lly Pvt. Ltd. All rights reserved.
        </msg>
    </footer>
</giftCards>
```

Schema Document

An XML schema document can be created through three methods which are Russian Doll, Salami Slice and Venetian Blind. For this XML document the Salami Slice method was selected.

The code of the XML Schema is displayed below.

Schema Code

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="giftCards">
        <xs:complexType>
            <xs:all>
                <xs:element ref="store"></xs:element>
                <xs:element ref="cards"></xs:element>
                <xs:element ref="footer"></xs:element>
            </xs:all>
        </xs:complexType>
    </xs:element>

    <xs:element name="store">
        <xs:complexType>
            <xs:all>
                <xs:element ref="nav"></xs:element>
                <xs:element ref="details"></xs:element>
                <xs:element ref="Slogan"></xs:element>
                <xs:element ref="URL"></xs:element>

            </xs:all>
        </xs:complexType>
    </xs:element>

    <xs:element name="nav">
        <xs:complexType>
            <xs:all>
```

```
            <xs:element ref="logo"></xs:element>
            <xs:element ref="option1"></xs:element>
            <xs:element ref="option2"></xs:element>
            <xs:element ref="option3"></xs:element>
            <xs:element ref="searchbar"></xs:element>

        </xs:all>
    </xs:complexType>
</xs:element>

<xs:element name="logo">
    <xs:complexType>
        <xs:all>
            <xs:element ref="image"></xs:element>
            <xs:element ref="logoText"></xs:element>
        </xs:all>
    </xs:complexType>
</xs:element>

<xs:element name="image">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute ref="id"></xs:attribute>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>

<xs:attribute name="id" type="xs:string"></xs:attribute>
<xs:element name="logoText" type="xs:string"></xs:element>

<xs:element name="option1" type="xs:string"></xs:element>
<xs:element name="option2" type="xs:string"></xs:element>
<xs:element name="option3" type="xs:string"></xs:element>
<xs:element name="searchbar">
    <xs:complexType>
        <xs:all>
            <xs:element ref="image"></xs:element>
        </xs:all>
    </xs:complexType>
</xs:element>
```

```
<xs:element name="details">
    <xs:complexType>
        <xs:all>
            <xs:element ref="name"></xs:element>
        </xs:all>
    </xs:complexType>
</xs:element>
<xs:element name="name" type="xs:string"></xs:element>
<xs:element name="Slogan" type="xs:string"></xs:element>
<xs:element name="URL" type="xs:string"></xs:element>


<xs:element name="cards">
    <xs:complexType>
        <xs:all>
            <xs:element ref="bestSeller"></xs:element>
            <xs:element ref="otherCards"></xs:element>
        </xs:all>
    </xs:complexType>
</xs:element>

<xs:element name="bestSeller">
    <xs:complexType>
        <xs:all>
            <xs:element ref="heading"></xs:element>
            <xs:element ref="items"></xs:element>
        </xs:all>
    </xs:complexType>
</xs:element>

<xs:element name="heading" type="xs:string"></xs:element>
<xs:element name="items">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="card" maxOccurs="3"></xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>

<xs:element name="card">
    <xs:complexType>
        <xs:sequence>
```

```
                <xs:element ref="img"></xs:element>
                <xs:element ref="cardName"></xs:element>
                <xs:element ref="type"></xs:element>
                <xs:element ref="cost"></xs:element>
                <xs:element ref="validity"></xs:element>
                <xs:element ref="availableNumber"
minOccurs="0"></xs:element>
                <xs:element ref="no_of_users"></xs:element>
                <xs:element ref="renew" minOccurs="0"></xs:element>
                <xs:element ref="discount" minOccurs="0"></xs:element>
                <xs:element ref="vat" minOccurs="0"></xs:element>
                <xs:element ref="refund" minOccurs="0"></xs:element>
                <xs:element ref="company" minOccurs="0"></xs:element>
            </xs:sequence>
            <xs:attribute ref="cardId"></xs:attribute>
        </xs:complexType>
    </xs:element>

    <xs:element name="img">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:string">
                    <xs:attribute ref="id"></xs:attribute>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="cardName" type="xs:string"></xs:element>
    <xs:element name="type" type="xs:string"></xs:element>
    <xs:element name="cost">
        <xs:complexType>
            <xs:simpleContent>
                <xs:extension base="xs:positiveInteger">
                    <xs:attribute ref="currency" fixed="$"
use="optional"></xs:attribute>
                </xs:extension>
            </xs:simpleContent>
        </xs:complexType>
    </xs:element>
    <xs:attribute name="currency" type="xs:string"></xs:attribute>

    <xs:element name="validity" type="xs:gYear"></xs:element>
```

```
    <xs:element name="availableNumber"
type="xs:positiveInteger"></xs:element>
    <xs:element name="no_of_users" type="xs:positiveInteger"></xs:element>
    <xs:element name="discount">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="discount_vouchers"
minOccurs="0"></xs:element>
            </xs:sequence>
            <xs:attribute ref="Available" default="true"
use="optional"></xs:attribute>
        </xs:complexType>
    </xs:element>

    <xs:element name="discount_vouchers" type="xs:string"></xs:element>
    <xs:attribute name="Available" type="xs:boolean"></xs:attribute>
    <xs:element name="vat" type="xs:string"></xs:element>


    <xs:element name="renew">
        <xs:complexType>
            <xs:all>
                <xs:element ref="Renew"></xs:element>
            </xs:all>
            <xs:attribute ref="option"></xs:attribute>
        </xs:complexType>
    </xs:element>
    <xs:element name="Renew" type="xs:string"></xs:element>
    <xs:attribute name="option">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="Yes"></xs:enumeration>
                <xs:enumeration value="No"></xs:enumeration>
                <xs:enumeration value="Will be available
soon"></xs:enumeration>

            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>

    <xs:element name="refund">
        <xs:complexType>
            <xs:all>
```

```
                <xs:element ref="Refund"></xs:element>
            </xs:all>
            <xs:attribute ref="choice"></xs:attribute>
        </xs:complexType>
    </xs:element>

    <xs:element name="Refund" type="xs:string"></xs:element>
    <xs:attribute name="choice">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="Yes"></xs:enumeration>
                <xs:enumeration value="No"></xs:enumeration>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>

    <xs:element name="company">
        <xs:complexType>
            <xs:all>
                <xs:element ref="Company"></xs:element>
            </xs:all>
            <xs:attribute ref="name"></xs:attribute>
        </xs:complexType>
    </xs:element>


    <xs:element name="otherCards">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="heading"></xs:element>
                <xs:element ref="items"
maxOccurs="unbounded"></xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="Company" type="xs:string"></xs:element>
    <xs:attribute name="name" type="xs:string"></xs:attribute>
    <xs:attribute name="cardId" type="xs:ID"></xs:attribute>

    <xs:element name="footer">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="storeName"></xs:element>
```

```
                <xs:element ref="address"></xs:element>
                <xs:element ref="telephone"></xs:element>
                <xs:element ref="msg1"></xs:element>
                <xs:element ref="follow"></xs:element>
                <xs:element ref="msg" maxOccurs="2"></xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="storeName" type="xs:string"></xs:element>
    <xs:element name="address">
        <xs:complexType>
            <xs:all>
                <xs:element ref="text"></xs:element>
            </xs:all>
        </xs:complexType>
    </xs:element>
    <xs:element name="text" type="xs:string"></xs:element>

    <xs:element name="telephone">
        <xs:complexType>
            <xs:all>
                <xs:element ref="text"></xs:element>
            </xs:all>
        </xs:complexType>
    </xs:element>

    <xs:element name="msg1" type="xs:string"></xs:element>
    <xs:element name="follow">
        <xs:complexType>
            <xs:all>
                <xs:element ref="text"></xs:element>
            </xs:all>
        </xs:complexType>
    </xs:element>
    <xs:element name="msg" type="xs:string"></xs:element>

</xs:schema>
```

Difference between schema and DTD

XML schema is widely known as XML Schema Definition (XSD). It is used to describe and validate the formation and framework of XML data. It can also be defined as the blueprint of an XML document.

DTD is the short form of Document Type Definition. A DTD defines the structure and component (elements and attributes) of an XML document. DTD helps to determine the validity and syntax of a document. The difference between DTD and Schema is;

Difference between Schema and DTD.

S.N.	Schema	DTD
1.	XML schema is known for utilizing an XML-based syntax.	DTDs are derived from SGML and have a unique form of syntax in compared to XML and XML schema.

2.	XML schema is used as the blueprint for an XML document.	DTDs are used to retain a level of compatibility with SGML for the application that would later want to convert SGML DTD into an XML DTD.
3.	Schema initially defines the datatypes for elements and attributes.	DTD does not support any data type.
4.	XML schemas support namespaces.	DTDs do not support namespaces.
5.	Schemas define the number and order of all the child elements.	DTD does not define the number and order of any child elements.
6.	An entity cannot be defined in an XML Schema.	An entity can be defined in XML schema.

7.	An XML schema can later be manipulated according to the XML document.	DTD cannot be altered according to the XML document.
8.	XML schema provides a more secured data communication between the sender and the receiver.	DTD does not provide a secured data communication because of which the data can be misunderstood by the receiver.
9.	Schemas linked to XML are extensible.	DTD is not extensible

Table 3: Difference between XML and Schema

Testing

1. Validating XML alone.

Test Number	**01**
Input	The XML document was copied and then pasted on https://www.xmlvalidation.com/ .
Expected Output	The XML document should validate without any errors.
Actual Output	The XML document was validated without any errors.
Test Output	The test was carried out successfully.

Table 4: Test 1 (Validating XML alone.)

Test Output

Before Validation

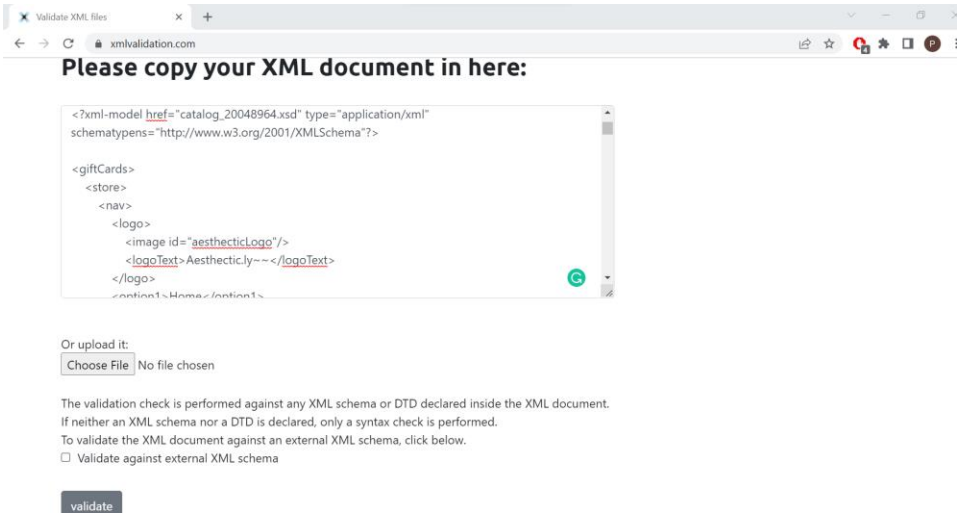

Figure 4: Test 1(Validating XML document 1)

After validation

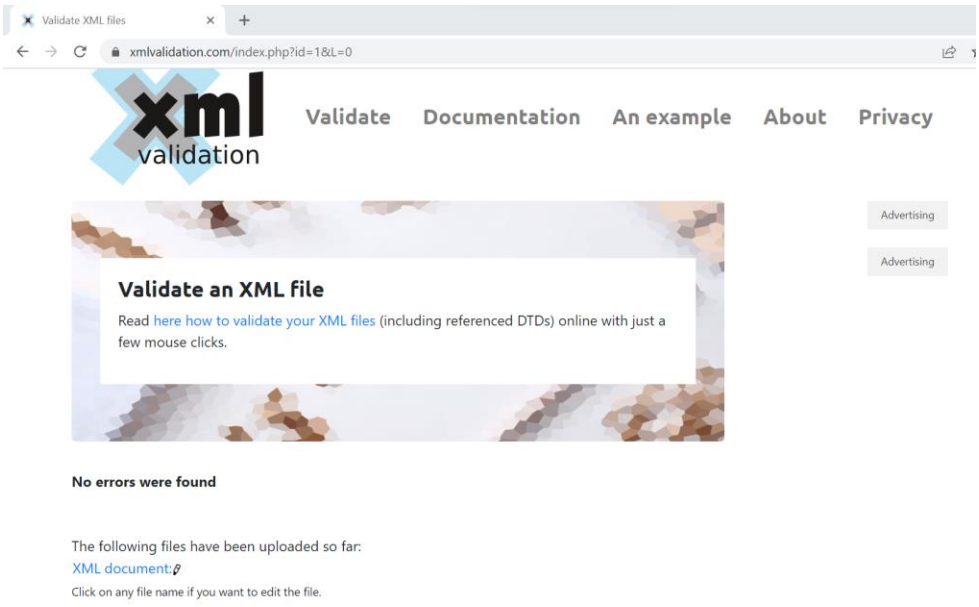

Figure 5: Test 1(Validating XML document 2)

2. Validating CSS alone.

Test Number	**02**
Input	The codes from catalog_20048964.css was copied to http://www.css-validator.org/#validate_by_input and the output was checked.
Expected Output	The CSS file should be verified without any errors.
Actual Output	The CSS file was verified without any error.
Test Output	The test was carried out successfully.

Table 5: Test 2(Validating Schema alone.)

Test Output

Before validating CSS

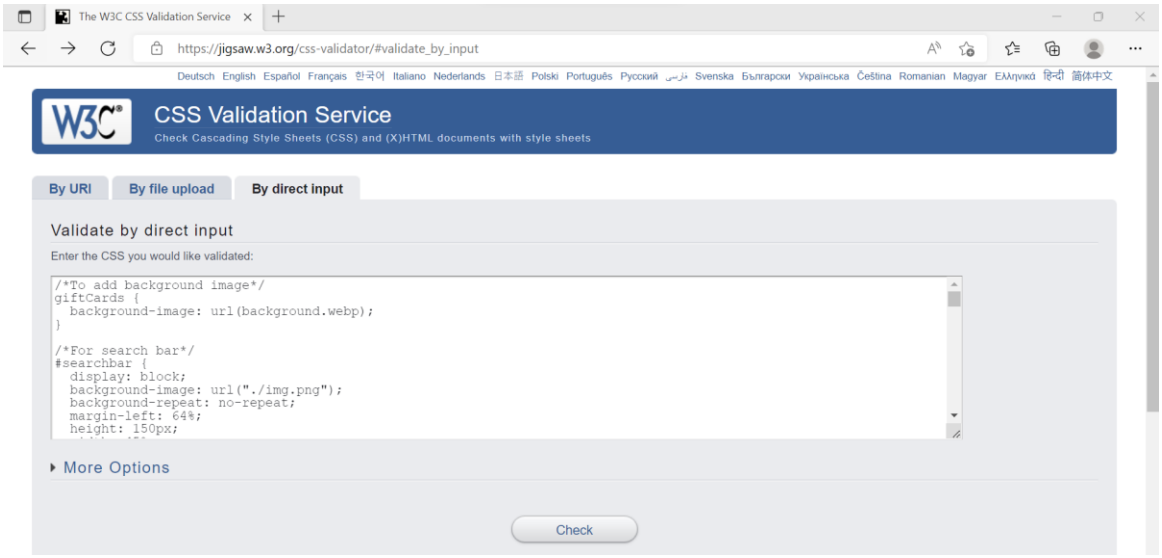

Figure 6: Test2 (Validating CSS 1)

After validating CSS

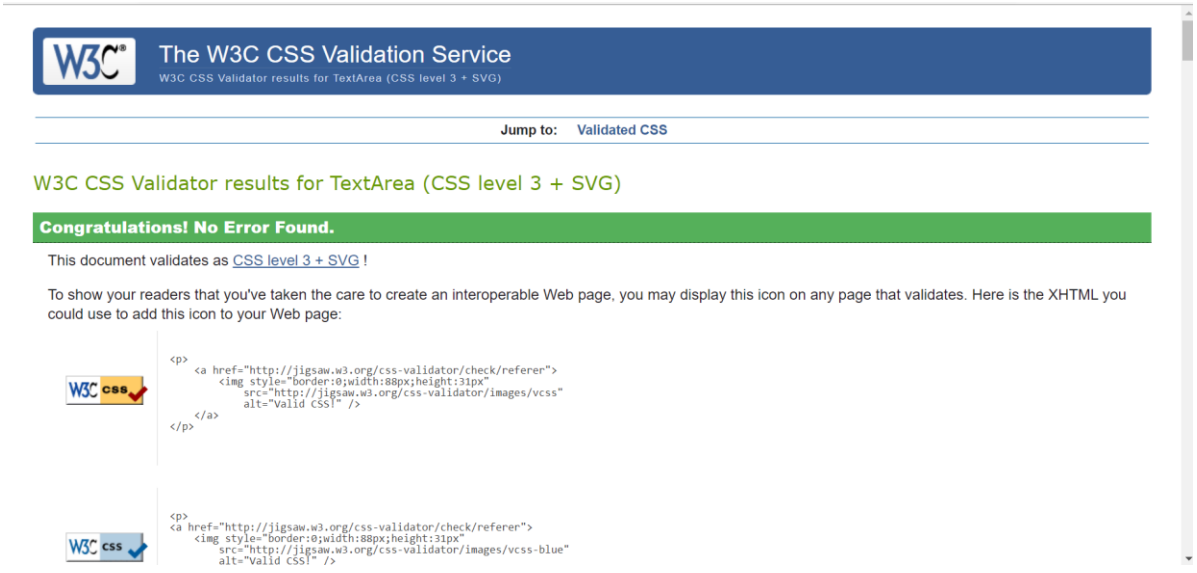

Figure 7: Validating CSS alone

3. Validating XML and Schema.

Test Number	**04**
Input	The code of XML document was pasted on https://www.xmlvalidation.com/ and then the checkbox "Validate against external schema was clicked and the code of schema was pasted.
Expected Outpt	No errors should be found on both the documents.
Actual Output	No errors were found on both the documents.
Test Output	The test was carried out successfully.

Table 6: Test 3(Validating XML document with Schema)

Test Output

Pasting the XML document.

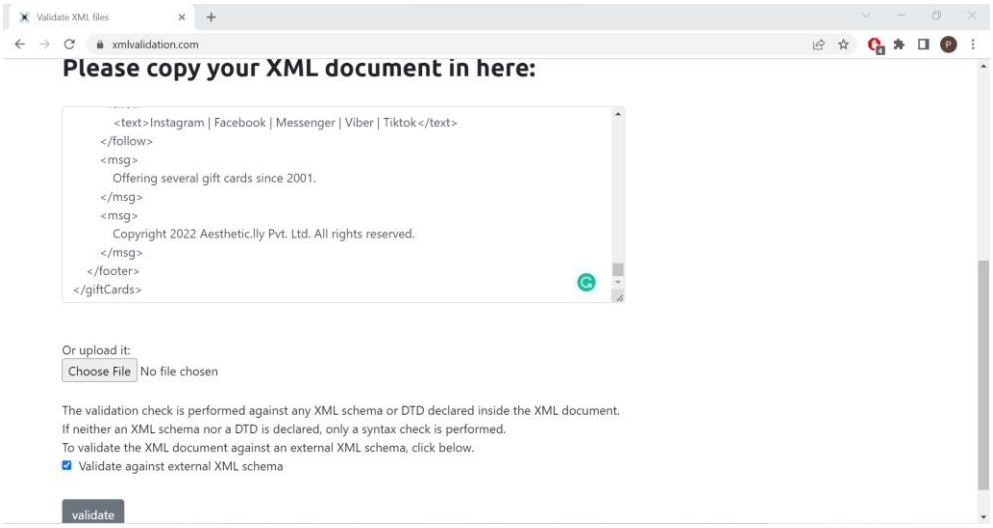

Figure 8: Test4(Pasting the XML document.)

Pasting the XML schema

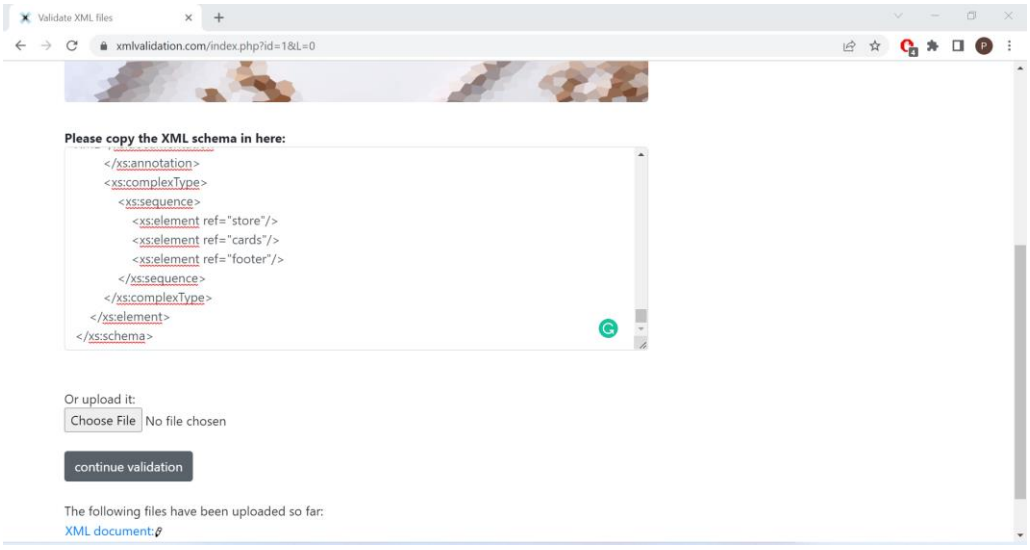

Figure 9: Test4 (Pasting the XML document.)

Validation of XML and Schema

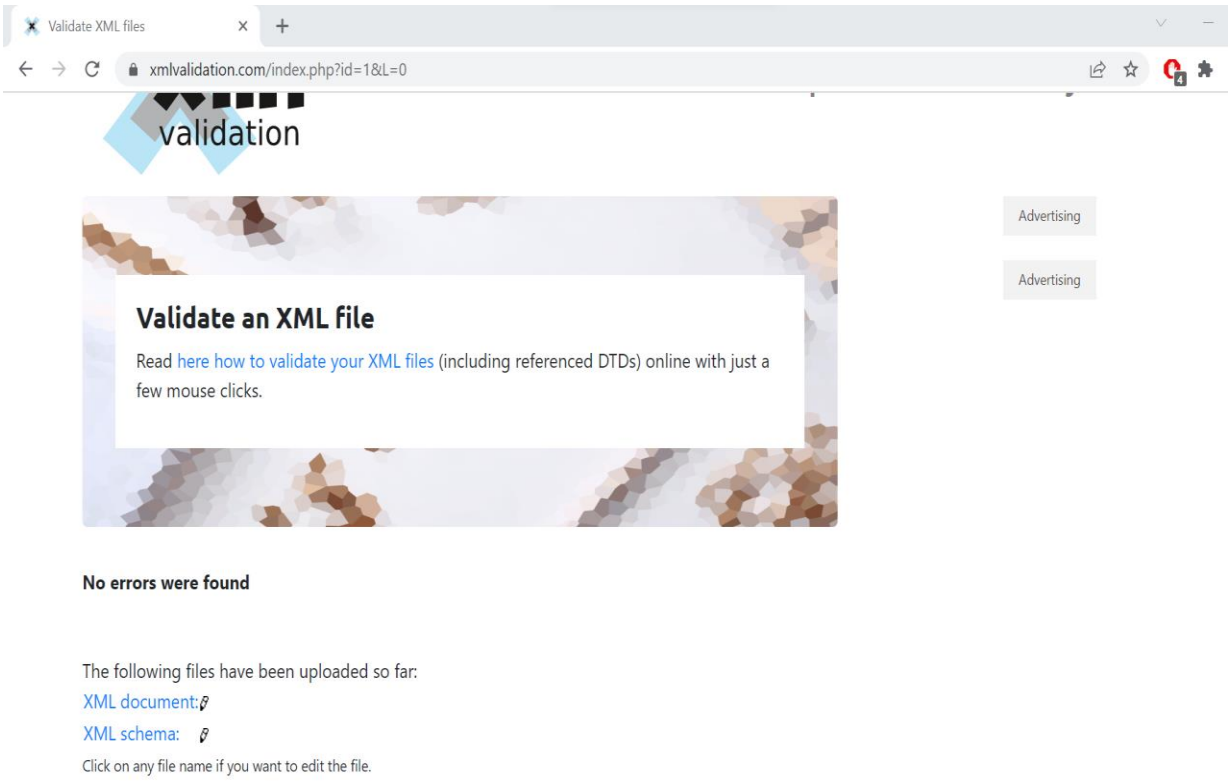

Figure 10: Test 4(Validation of XML and Schema)

4. Compiling the XML code along with CSS.

Test Number	**04**
Input	The file catalog_20048964.css was deleted from the folder and then the XML document was compiled. The output was observed.
Expected Output	The program will run smoothly but the formatting of XML document will be affected.
Actual Output	The program ran smoothly but the data like images, card names, store details, etc. was not represented in a proper manner.
Test Output	The test was carried out successfully.

Table 7: Test 4(Compiling XML code without CSS)

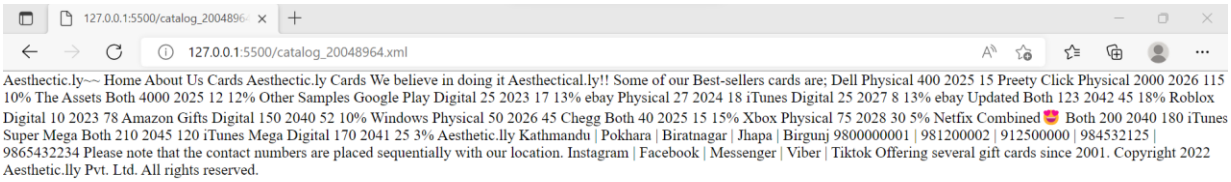

Figure 11: Task 4(XML code without CSS)

5. Compiling the XML code without CSS.

Test Number	**05**
Input	The file catalog_20048964.css was restored, and output was observed by compiling the XML document.
Expected Output	The program will run smoothly and display all the data clearly with proper formatting.
Actual Output	The program will ran smoothly and display all the data clearly with proper formatting.

Test Output	The test was carried out successfully.

Table 8: Test 5(Compiling XML code without CSS)

Test Output

Compiling with CSS.

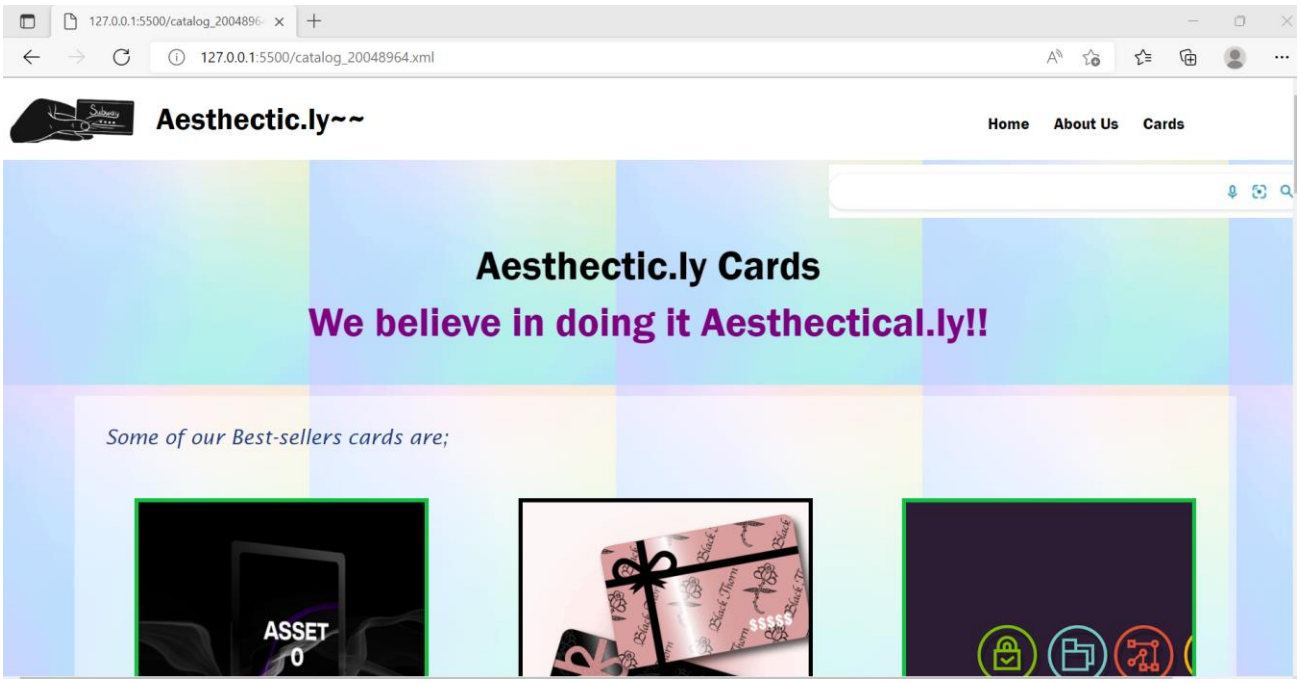

Figure 12: Test 5(Compiling the code with CSS)

6. Analyzing the hover effect of CSS.

Test Number	**06**
Input	Mouse was dragged close to a card and the result was observed.
Expected Output	The image of the card should get magnified.
Actual Output	The image of the card was magnified.
Test Output	The test was carried out successfully.

Table 9: Test 6(Analyzing the hover effect of CSS.)

Test Output

Before clicking on the card

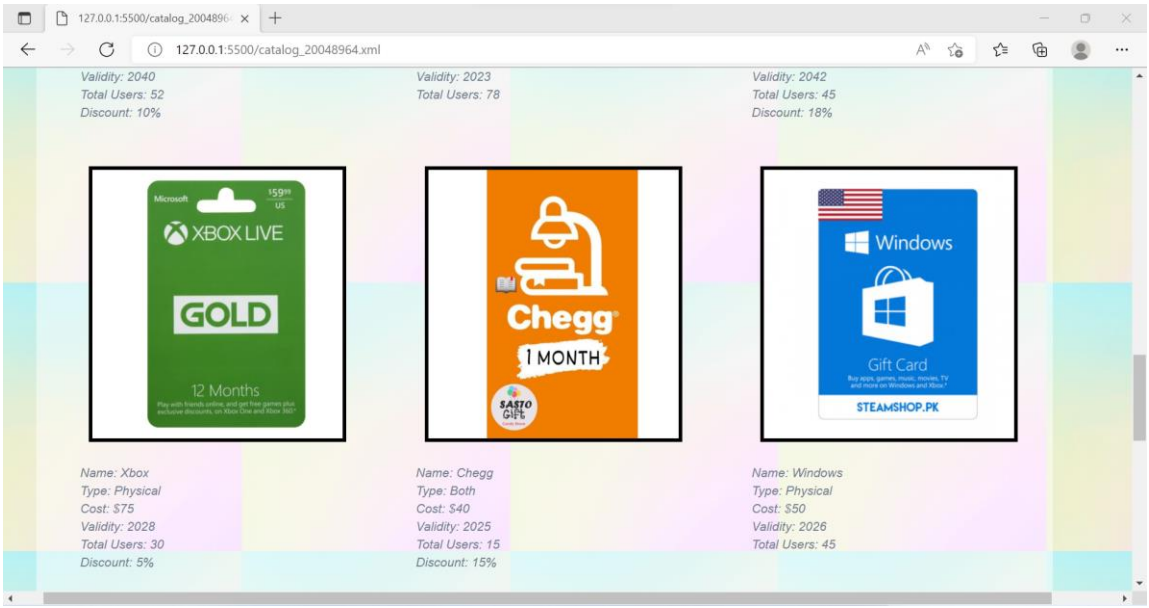

Figure 13: Test 6 (To show the hover effect.)

After clicking on the card

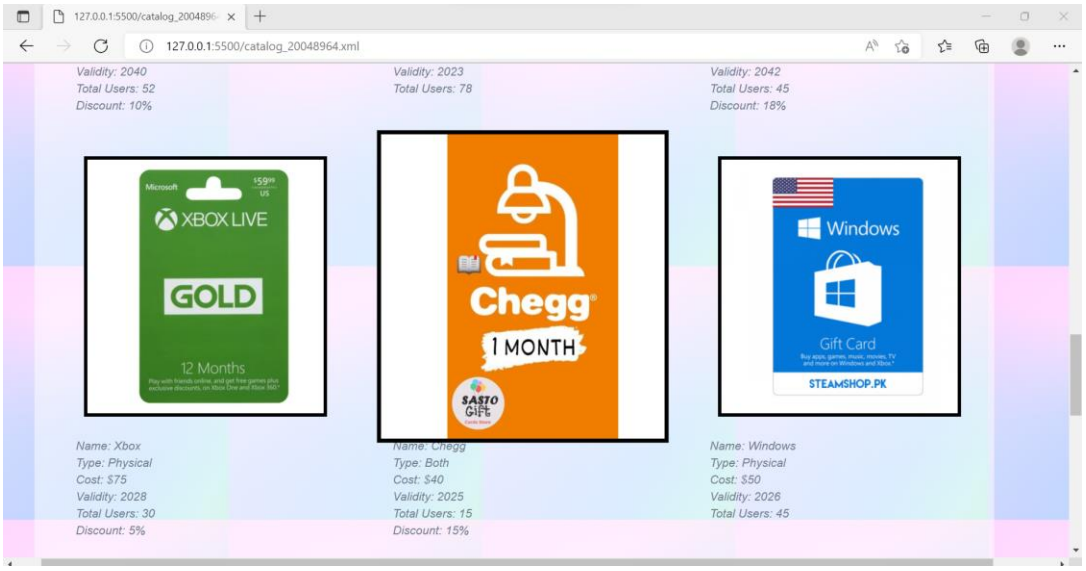

Figure 14: Test 6 (To show the hover effect on CSS)

Development of Coursework

To start this coursework, a thorough the coursework requirements was noted and a thorough research was done about the topic. All the topics like XML, XML schema, DTD, HTML and XHTML was done. An appropriate content and method for each requirement was selected. Like for the development of XML schema the salami slice method was selected amongst Russian Doll, Salami Slice and Venetian Blind. Similarly, for the structure of website XML was selected as the markup language. And XHTML was not used throughout the coursework. After this, the tree diagram was created by selecting the appropriate element and tags according to the requirement.

After this an XML file was created, and all the declaration of prolog elements were done. The root element was named as giftCards which had two child element store cards and footer. The child element store acts as the parent element to nav, details, and Slogan. The primary function of this element is to store the details of the store, a slogan for the store, and contribute to other components like the navigation bar and image for the logo. The child element of the root element (giftCards) cards acts as the parent element of bestSeller and otherCards. This element is responsible to store the details of the cards in the store like card id, cost, discount available, company name, validity date, type of the card, and image of the card. Finally, the last direct child element of the root element giftCards is the footer. This element is used to create a platform to store the necessary details about the contacts of the store.

After defining all the necessary requirements, the optional elements were selected which are renew, refund, tax, vat, company name and others. Again, the attribute was selected which includes the information about company name, discount availability, company name for the cards, renew and refund options.

After defining the structure of the website, a CSS file was created to define the formatting of the website. Different effects like the hover effect, float and other properties was added to the XML document. The comparison of the document before

linking the CSS document and after defining the CSS document is represented below.

Before adding the CSS file.

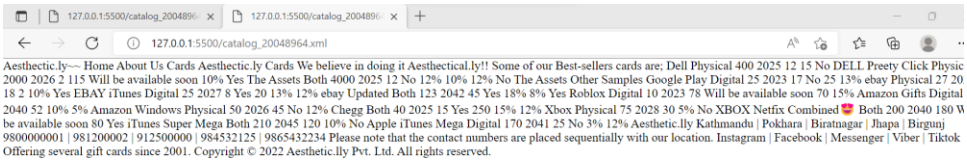

Figure 15: Before adding the CSS file

After adding the CSS file

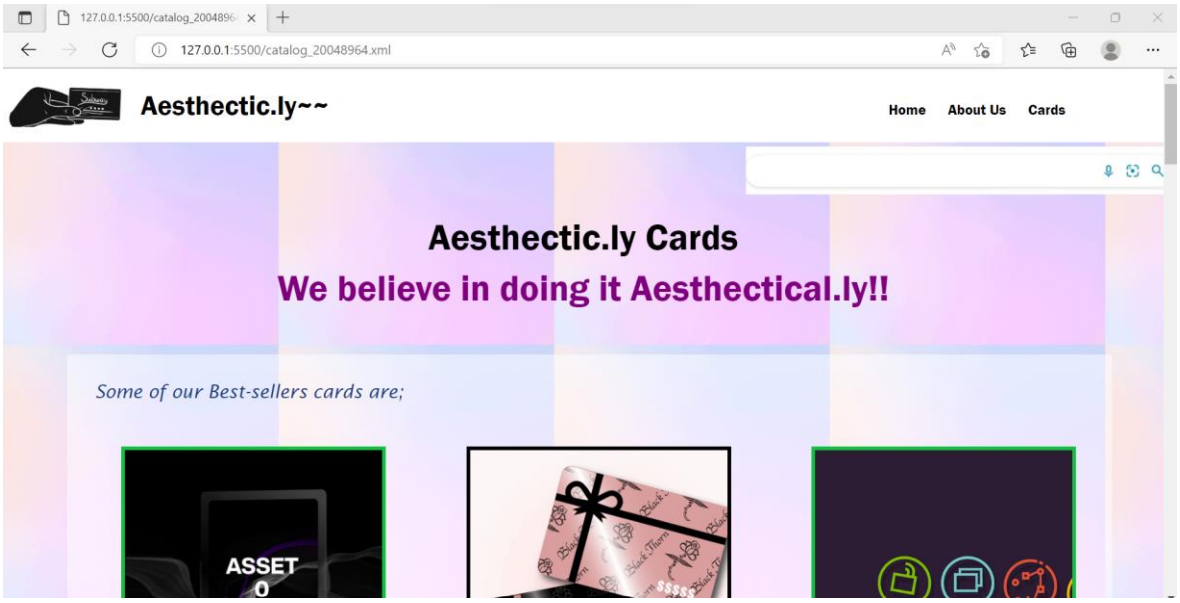

Figure 16: After linking CSS with XML document

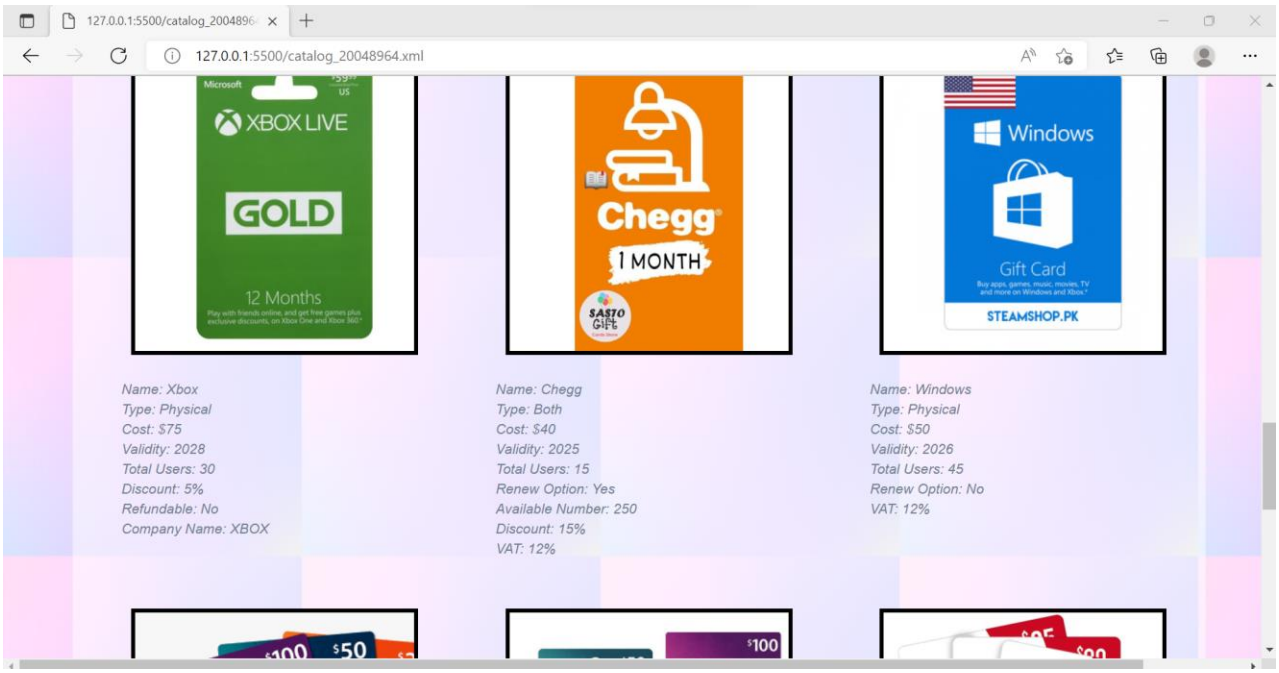

Figure 17: After linking the XML document with CSS.

Along with adding the CSS file, an XML schema was also added. The schema was used for XML validation. For XML schema, the Salami Slice method was selected. After this report was prepared which included the information about testing, coursework development, critical analysis, list of all the data and attributes and many more details.

The errors were debugged through the help of teachers and own research. Additional research was done after thoroughly learning about the topics from the lecture slides and lab tasks.

Use of Tools and Methods

The requirements of the coursework were kept in mind throughout the development of the coursework. The code was written in Visual Studio Code and Snipping tool was used to take the screenshot of the evidence and codes. For the construction, of the tree diagram draw.io was used. Research was done from various websites and the lecture and tutorial materials was studied effectively.

Critical Analysis

This assignment is the result of the result of consistent effort and dedication which has led to a fruitful result and fulfillments of all the requirements of the coursework. It contains of an XML document that is used to specify the structure and the content of the browser, a CSS file that is used for the formatting of the document, an XSD file for describing and validating the contents of the XML document and a report which consists of the overall documentation and testing of all the three files.

This project has fulfilled all the requirements of the coursework. The XML document consists of all the details and tags that were asked to be included in the coursework like;

7. About 36 unique data (XML tags) is included.
8. Fifteen unique cards with unique data.
9. Six unique attributes have been added that are id (for image), cardId (for card), option (renew), choice (refund), Available (discount status) and name (of the company).
10. Seven optional data have been added that includes renew, companyName, refundable, availableNumber, discount and vat.

A CSS file was created for providing the XML document with a proper formatting that meets the following requirements.

- Use of logo on the website.
- Use of flex as floating box.
- Displaying 15 images of cards with borders.

The requirements met by the XSD (schema) file are;

- Applicable and user-driven data are included.
- All the elements like price, validity, etc. has been properly defined.

The report includes the vital information like tree diagram, development of the coursework, all the testing, list of the elements and attributes, difference between an XML schema and DTD, evidence of the development of coursework and many more crucial information.

The limitation of the website or the coursework are;

- Confusion to select the best method for the construction of an XML schema.
- The CSS file was a little difficult to work with.
- Confusion about correct use of CSS properties.
- Adding optional data created a little confusion.

Conclusion

This is the second coursework of Emerging Programming Platforms and Technologies (Module Code: CS5004NI). This coursework was given out to students to enhance their knowledge about XML, datatypes, various types of XML schema, elements and attributes and CSS. This coursework was completed through the guidance of teachers and an enormous amount of research.

For the submission of this coursework, the folders including the images of the cards, XML documentation, an XML schema (xsd file), a CSS file and this report (in pdf format) was submitted. The coursework helped to strengthen the knowledge relate to other components of XML like XHTML, Schema proper way of documentation and many more. The coursework would not have been effective and running smoothly without the guidance of teachers and the lecture and tutorial materials provided in the classroom.

The result of the coursework includes a website of a gift card shop named Aesthetic.lly. The website includes about fifteen sample gift cards that contains various details related to the cards along with the details of the store.

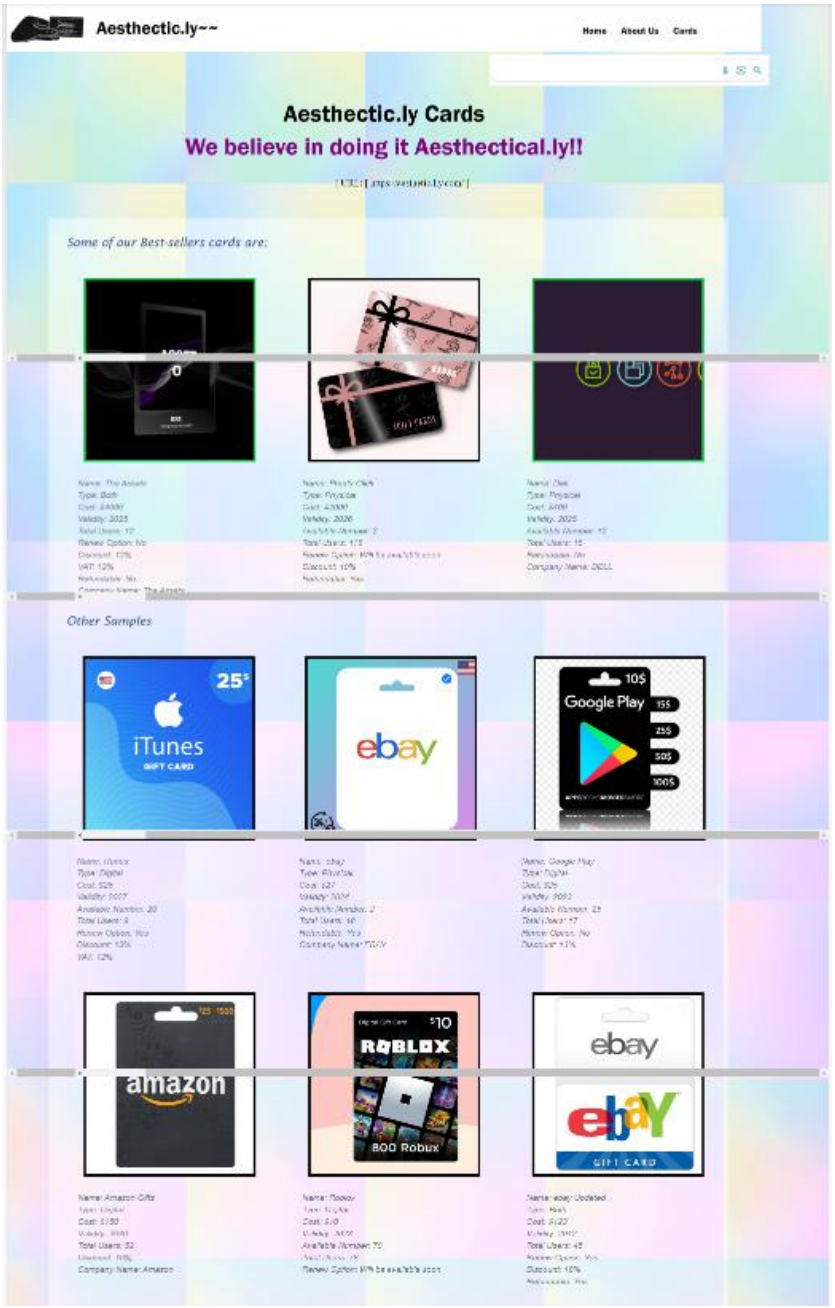
Aesthetic.ly~~
Home
About Us
Cards
Aesthetic.ly Cards
We believe in doing it Aesthetical.ly!!
Some of our Best-sellers cards are:
Other Samples
iTunes
GIFT CARD
ebay
Google Play
10$
15$
25$
50$
100$
amazon
ROBLOX
800 Robux
ebay
ebay
GIFT CARD

Figure 18: Image of the full website

References

Contributors, M., 2022. *mdn web docs*. [Online]

Available at: https://www.microfocus.com/documentation/extend-acucobol/925/BKITITNONVS004.html

[Accessed 2 May 2022].

Micro Focus Team, n.d. *XML Documents.* [Online]

Available at: https://www.microfocus.com/documentation/extend-acucobol/925/BKITITNONVS004.html

[Accessed 29 April 2022].

Thivnet, P., 2009. *Stack Overflow*. [Online]

Available at: https://stackoverflow.com/questions/1544200/what-is-difference-between-xml-schema-and-dtd#:~:text=Differences%20between%20an%20XML%20Schema,namespaces%20while%20DTD%20does%20not

[Accessed 3 May 2022].

W3Schhols Members, n.d. *W3Schhols.* [Online]

Available at: https://www.w3schools.com/cssref/default.asp

[Accessed 4 May 2022].

W3Schools Members, 2022. *W3 schools*. [Online].

W3Schools Members, n.d. *W3Schools*. [Online]

Available at:

https://www.w3schools.com/xml/schema_intro.asp#:~:text=The%20purpose%20of%20an%20XML,and%20order%20of)%20child%20elements

[Accessed 3 May 2022].

W3Schools Members, n.d. *W3Schools*. [Online]

Available at: https://www.w3schools.com/css/

[Accessed 2 May 2022].