



**slington college**  
(इस्लिङ्टन कलेज)

**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: Bishal Thapa**

**London Met ID:20048874**

**College ID: NP01CP4S210309**

**Assignment Due Date: May 5, 2022**

**Assignment Submission Date: May 5, 2022**

**Word Count (Where Required):5000**

*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

1. Introduction.....	1
1.1. Extensible Markup Language(XML) .....	1
1.2. XSD (Xml Schema Definition).....	2
1.3. Cascading Style Sheets (CSS).....	3
2. XML Document.....	4
2.1. List of Data .....	4
2.1.1. List of Elements .....	4
2.1.2. List of Attributes .....	6
2.2. Tree Diagram .....	6
2.3. XML Code .....	7
3. Schema Document.....	15
4. Difference between Schema and DTD .....	18
5. Testing.....	19
5.1. Test 1 .....	20
5.2. Test 2 .....	22
5.3. Test 3 .....	23
5.4. Test 4 .....	25
5.5. Test 5 .....	26
6. Coursework Development.....	28
7. Critical Analysis .....	31
8. Conclusion.....	39
9. References .....	40
10. Appendix .....	41

## Table of Figures:

Figure 1: XML .....	2
Figure 2: The picture of XSD .....	3
Figure 3: The picture of CSS.....	4
Figure 4: Tree Diagram .....	7
Figure 5: Validating the XML file on validator .....	20
Figure 6: Validating XSD file on validator .....	21
Figure 7: The result of XML and XSD file .....	22
Figure 8: Before removing card_status element.....	23
Figure 9: After removing card_status element.....	23
Figure 10: Unlinking the Css file .....	24
Figure 11: The output after unlinking CSS file .....	24
Figure 12: Linking the CSS file.....	25
Figure 13: The output after linking the CSS file. ....	25
Figure 14: Before hovering the content.....	26
Figure 15: After hovering the content.....	27
Figure 16: Hovering the text content. ....	27
Figure 17: Using VS code text editor for developing the project .....	28
Figure 18: Creating tree diagram by using draw.io .....	29
Figure 19: Developing the document file by using MS word .....	30
Figure 20: Validating the XML and XSD by using online validator .....	30
Figure 21: critical analyzing the problem 1 .....	32
Figure 22: Critical analyzing the solution 1 .....	32

Figure 23: critical analyzing the problem 2 .....	33
Figure 24: critical analyzing the solution 2 .....	33
Figure 25: critical analyzing the problem 3 .....	34
Figure 26: critical analyzing the solution 3 .....	34
Figure 27: Critical analysis of the problem 4 .....	35
Figure 28: Critical analysis and output of the problem 4 .....	35
Figure 29: Critical analysis of the solution 4 .....	36
Figure 30: Critical analysis of the solution 4 output .....	36
Figure 31: Critical analysis of the problem 5 .....	37
Figure 32: Critical analysis of the problem output 5 .....	37
Figure 33: Critical analysis of the solution 5 .....	38
Figure 34: Critical analysis of the solution output 5 .....	38

#### Table of Tables:

Table 1: List of elements .....	5
Table 2: List of Attributes .....	6
Table 3: Difference Between Schema and DTD .....	19
Table 4: Validating XML file .....	20
Table 5: Validating XML and XSD file with result .....	21
Table 6: Checking the optional element .....	22
Table 7: Testing the xml file without using css file .....	23
Table 8: Testing after linking CSS file .....	25
Table 9: Testing hover effect .....	26

## 1. Introduction

The coursework focuses on using XML, CSS, and schema to design a gift shop. We are given the scenario and told to design any gift cards we want using the technologies listed above. From all the information which was provided in the question I have created a website that is used to sell gift cards in London named as Steak 44. The store sells various gift cards in two formats: physical and digital. The website contains several important details about stores and cards. The store name, address, phone number, and website are all listed in the store information. The name of the card, its type, amount, payment method, genre, card status, production company, discount voucher, shipping cost, shipping address, tax, number of users, validation duration, Card Code, and description are also displayed in the Card section.

The coursework was made possible by combining several technologies, including XML, CSS, and XSD. A tree diagram has also been created to show each element's and attribute's properties.

### 1.1. Extensible Markup Language(XML)

Full extensible markup language, or XML, is a document formatting language that is used on some World Wide Web pages. Because HTML, the most common format for Web pages, does not allow the definition of new text elements, or is not extensible, XML was created in the 1990s. XML is a web-friendly version of SGML. DTDs are used in XML to define document types and the meanings of tags used in them, just like they are in SGML. XML follows parsing conventions, such as document entities having both a beginning and an ending tag, and XML supports more types of hypertext links than HTML, such as bidirectional links and links relative to a location (Hemmendinger, 2008).

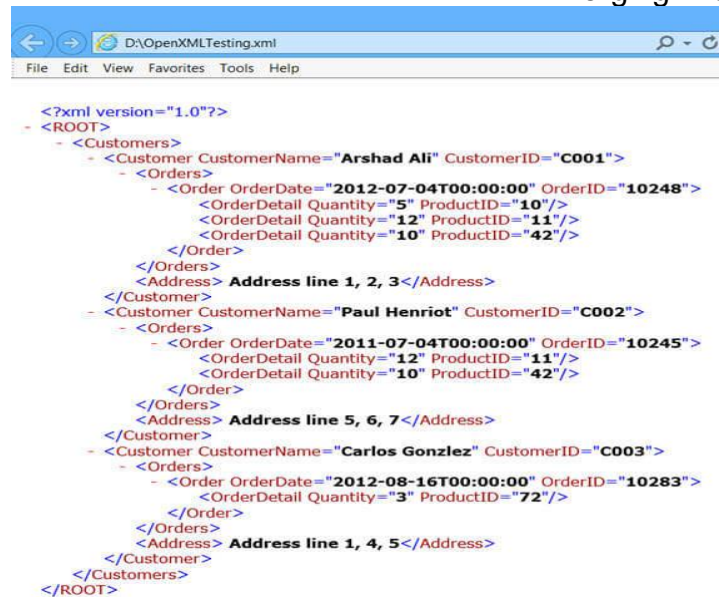


Figure 1: XML

Source: [https://www.mssqltips.com/tipimages2/2899\\_img5.jpg](https://www.mssqltips.com/tipimages2/2899_img5.jpg)

A short list of XML usage says it all :-

- XML can simplify the creation of HTML documents for large web sites by working behind the scenes.
- Information can be exchanged between organizations and systems using XML.
- Databases can be offloaded and reloaded using XML.
- XML can be used to store and organize data, allowing you to tailor your data management needs.
- XML and style sheets can easily be combined to create almost any desired output.
- An XML document can be used to express almost any type of data (tutorialspoint, 2022).

## 1.2. XSD (Xml Schema Definition)

The XML Schema Definition (XSD) language is the current standard schema language for all XML documents and data. The World Wide Web Consortium (W3C) released XSD in its version 1.0 format on May 2, 2001. The XML Schema definition language (XSD) allows you to define the structure and data types of XML documents. An XML Schema defines the elements, attributes, and data types that conform to the World Wide Web Consortium

(W3C) XML Schema Part 1: Structures. Recommendation for the XML Schema Definition Language. W3C XML Schema Part 2: Datatypes Recommendation is a recommendation for defining data types in XML schemas. The XML Schema Reference (XSD) is based on the W3C 2001 Recommendation specifications for Datatypes and Structures. The schema element includes type definitions, simple Type and complex Type elements, attribute and element declarations, and attribute and element declarations. XML Schema allows the definition of new data types using the simple Type and complex Type elements, in addition to its built-in data types such as integer, string, and so on.

- simple Type: A value type definition that can only be used as the content text Only of an element or attribute. This data type does not support elements or attributes.
- Complex Type is a type of definition for elements that can have attributes and elements. This data type can have elements as well as attributes (Microsoft Build, 2016).



```
<!-- Schema Components -->
<xs:complexType name="baseComponent">
  <xs:complexContent> [29 lines]
</xs:complexType>
<xs:complexType name="componentWithFacets">
  <xs:complexContent>
    <xs:extension base="baseComponent"> [3 lines]
  </xs:complexContent>
</xs:complexType>
<xs:element name="schema">
  <xs:complexType>
    <xs:complexContent>
      <xs:extension base="baseComponent">
        <xs:attribute name="type" use="required"
```

Figure 2: The picture of XSD

Source: [https://www.oxygenxml.com/img/xsd\\_highlight\\_occurences.png](https://www.oxygenxml.com/img/xsd_highlight_occurences.png)

### 1.3. Cascading Style Sheets (CSS)

Cascading Style Sheets, affectionately alluded to as CSS, is a straightforward plan language proposed to work on the way toward making site pages adequate. CSS handles the look and feel some portion of a page. Utilizing CSS, you can handle the shade of the

content, the style of textual styles, the separating between sections, how segments are estimated and spread out, what foundation pictures or shadings are utilized, format plans, varieties in show for various gadgets and screen measures just as an assortment of different impacts. CSS is not difficult to learn and see however it gives incredible authority over the introduction of a HTML archive. Most normally, CSS is joined with the mark-up languages HTML or XHTML (tutorialspoint, 2022). Some benefits of CSS are such as:

- Web pages load faster and consume less bandwidth.
- Greater consistency in design
- CSS enables you to place your element anywhere on the page.
- CSS style sheets enable the user to personalize the webpage.
- CSS enables web pages to be completely consistent (Alexa, Apr 11 2008).



Figure 3: The picture of CSS

Source: <https://www.skyminds.net/css-remplir-le-contenu-dune-div-de-maniere-fluide/>

## 2. XML Document

### 2.1. List of Data

#### 2.1.1. List of Elements

List of Elements	Element Type	Element data	Occurrence
cardStore	complexType	Child elements	Only once
storeDetails	complexType	Child elements	Only once
storeName	simpleType	String	Only once

Logo	complexType	Empty element	Only once
Address	simpleType	String	Only once
Contact	complexType	String	Only once
Email	simpleType	String	Only Once
Telephone	simpleType	String	Only once
Fax	simpleType	NMTOKEN	Only once
url	simpleType	String	Only once
giftCards	complexType	Child elements	Only once
Cards	complexType	Child elements	One or more
Card_Cover	complexType	Empty element	Only once
Name	simpleType	String	Only once
Card_Type	simpleType	String	Only once
Card_amount	complexType	String	Only once
Payment_Method	simpleType	String	Only once
Card_status	simpleType	String	Zero or one
Genre	simpleType	String	Zero or one
Production_company	simpleType	String	Zero or one
validationDuration	simpleType	String	Only once
shippingCost	simpleType	String	Zero or one
shippingAddress	simpleType	String	Zero or one
Discount_Voucher	simpleType	String	Zero or one
Card_code	simpleType	String	Zero or one
Tax	simpleType	String	Zero or one
noOfUsers	simpleType	String	Only once
Description	simpleType	String	Zero or one
Footer	simpleType	string	Only once

Table 1: List of elements



## 2.1.2. List of Attributes

List of Attributes	Data Type	Default	Fixed	Use
Description	String	-	1	Required
Id(logo_ID)	String	1	-	Required
Id (card_id)	String	1	-	Required
Id(card_Cover)	ID	1	-	Required
On_stock	String	1	-	Required

Table 2: List of Attributes

## 2.2. Tree Diagram

A tree diagram is a new management planning tool that shows the hierarchy of tasks and subtasks required to achieve a goal. The tree diagram begins with a single item that divides into two or more, each of which divides into two or more, and so on. With a trunk and multiple branches, the finished diagram resembles a tree. It's a technique for breaking down broad categories into finer and finer levels of detail. Creating a tree diagram allows you to move your thoughts from generalities to specifics in a step-by-step manner (ASQ, 2022).

In this coursework, cardStore is a root element followed by its three child elements: storeDetails, giftCards and Footer.

- storeDetails has five elements which contains information about gift card shop i.e., StoreName, logo, contact, address and url. In here contact contains three more child elements such as email, telephone, and fax. And at least logo has one attribute named as "id".
- giftCards has one child element which occurs multiple times named cards, '+' sign in the tree diagram represents its reoccurring property. Cards have one attribute "id". Cards contain 16 child elements out of 9 are optional elements meaning the placement of those elements are not mandatory. Optional elements include card\_status, Genre, production\_company, shippingCost, shippingAddress, discount\_Voucher, card\_Code, tax and Description. These optional elements are represented by the '?' sign.
- Footer has no child elements.

The tree diagram of this project is followed as below:-

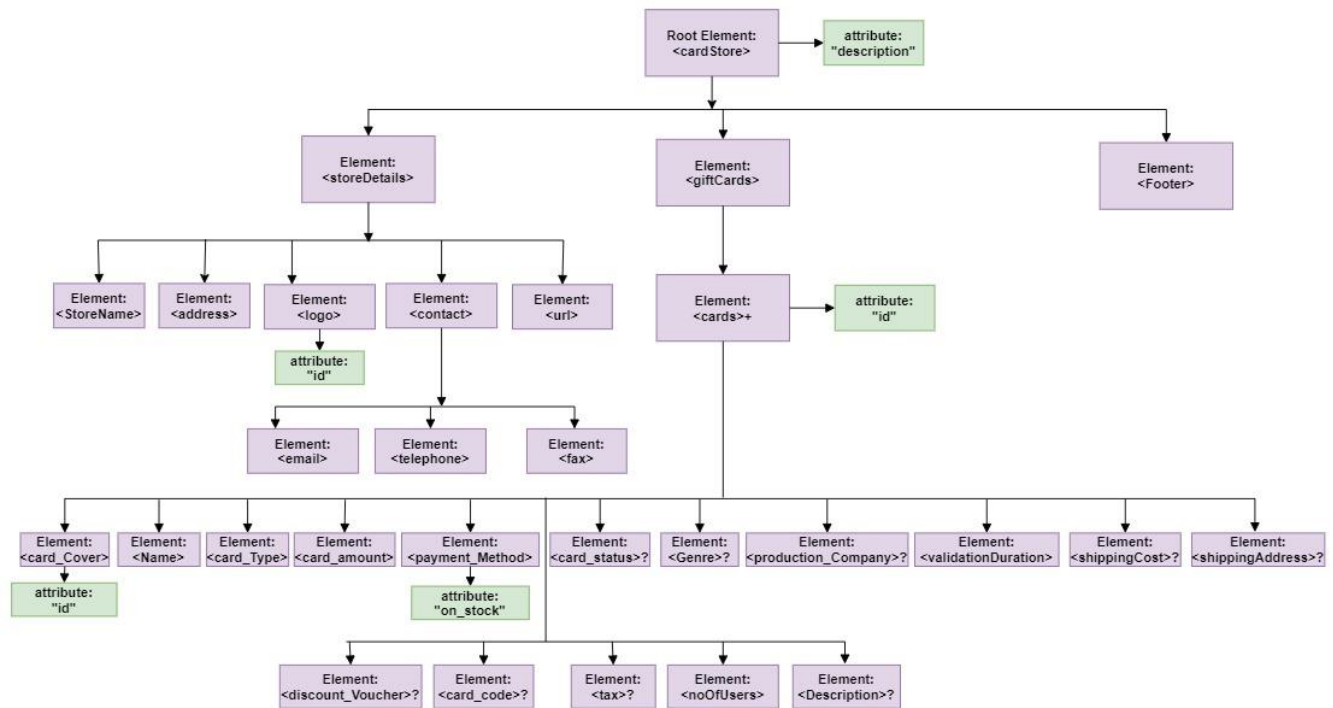


Figure 4: Tree Diagram

### 2.3. XML Code

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!--
  Author: Bishal Thapa
  Date: 03/05/2022
  Filename: catalog_20048874.xml
-->

<?xml-stylesheet href="catalog_20048874.css"?>
<!-- cardStore is the root element-->
<cardStore description="Gift Card Store in London"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="catalog_20048874.xsd" >
  <!-- cardStore is primarily divided into three child elements storeDetails, giftCards
  and Footer-->
  <!-- storeDetails stores storeName, Logo, address, contact and url-->

```

```
<storeDetails>
  <storeName>Steak 44</storeName>
  <!-- Logo of the company is incorporated in above element is referenced in css
file-->
  <!-- Also this element contains the address, contact and url -->
  <logo id ="Image"/>
  <address>London, United Kingdom</address>
  <!-- There are three child elements inside the contact element.-->
  <contact>
    <email>steak44@gmail.com</email>
    <telephone>(602) 271-4400</telephone>
    <fax> 02-123-5897852</fax>
  </contact>
  <url>uk.steak44.com</url>
</storeDetails>
<!-- cardStore has many giftCards child elements-->
<giftCards>
  <!-- giftCards element has cards child element. These child element includes
further information regarding particular gift card-->
  <cards id="card_1">
    <card_Cover id="card_xbox"></card_Cover>
    <Name>XBOX GOLD</Name>
    <card_Type>Digital</card_Type>
    <card_amount on_stock="yes">$50</card_amount>
    <payment_Method>Paypal, Google Pay</payment_Method>
    <Genre>Gaming</Genre>
    <validationDuration>6 months</validationDuration>
    <shippingCost>$5</shippingCost>
    <shippingAddress>London</shippingAddress>
    <discount_Voucher>20% off from DNB Bank</discount_Voucher>
    <Card_code>2564</Card_code>
    <tax>5 %</tax>
    <noOfUsers>120</noOfUsers>
  </cards>
  <cards id="card_2">
    <card_Cover id="card_itunes"></card_Cover>
    <Name>Itunes Gift Card</Name>
```

```

    <card_Type>Physical</card_Type>
    <card_amount on_stock="yes">$100</card_amount>
    <payment_Method>Credit Card, Cash, Bill Pay</payment_Method>
    <card_status>Active</card_status>
    <Genre>Music, Movie, Apps</Genre>
    <production_Company>Apple</production_Company>
    <validationDuration>10 months</validationDuration>
    <noOfUsers>554</noOfUsers>
    <Description>iTunes gift card is used to make purchases in the iTunes
Store without using a credit card.</Description>
  </cards>
  <cards id="card_3">
    <card_Cover id="card_hulu"></card_Cover>
    <Name>Hulu Gift Card</Name>
    <card_Type>Digital</card_Type>
    <card_amount on_stock="yes">$30</card_amount>
    <payment_Method>Google Pay, Paypal, Samsung Pay</payment_Method>
    <validationDuration>10 months</validationDuration>
    <shippingCost>$10</shippingCost>
    <shippingAddress>Liverpool</shippingAddress>
    <discount_Voucher>10% off from The Bank of London.</discount_Voucher>
    <Card_code>3658</Card_code>
    <tax>2 %</tax>
    <noOfUsers>550</noOfUsers>
  </cards>
  <cards id="card_4">
    <card_Cover id="card_amazon"></card_Cover>
    <Name>Amazon Gift Card</Name>
    <card_Type>Digital</card_Type>
    <card_amount on_stock="no">$30</card_amount>
    <payment_Method>Google Pay, Paypal</payment_Method>
    <Genre>Online Shopping</Genre>
    <validationDuration>1 year</validationDuration>
    <shippingCost>$5</shippingCost>
    <shippingAddress>Manchester</shippingAddress>
    <discount_Voucher>15% off from DNB Bank </discount_Voucher>
    <Card_code>9878</Card_code>
  </cards>

```

```
<tax>8 %</tax>
<noOfUsers>558</noOfUsers>
</cards>
<cards id="card_5">
  <card_Cover id="card_razer"></card_Cover>
  <Name>Razer Gold</Name>
  <card_Type>Digital</card_Type>
  <card_amount on_stock="yes">$500</card_amount>
  <payment_Method>Google Pay, Samsung Pay</payment_Method>
  <Genre>Gaming</Genre>
  <validationDuration>2 year</validationDuration>
  <shippingCost>$10</shippingCost>
  <shippingAddress>Hull</shippingAddress>
  <discount_Voucher>55% off from Doha Bank</discount_Voucher>
  <Card_code>5689</Card_code>
  <tax>2%</tax>
  <noOfUsers>154</noOfUsers>
</cards>
<cards id="card_6">
  <card_Cover id="card_valentine"></card_Cover>
  <Name>Valentine's Gift</Name>
  <card_Type>Physical</card_Type>
  <card_amount on_stock="yes">$50</card_amount>
  <payment_Method>Credit Card, Cash</payment_Method>
  <card_status>Active</card_status>
  <Genre>Netflix, Uber</Genre>
  <production_Company>Shop Me</production_Company>
  <validationDuration>1 year</validationDuration>
  <discount_Voucher>20% off from Danske Bank</discount_Voucher>
  <noOfUsers>157</noOfUsers>
  <Description>This card is specially made for couples in the occasion
Valentine Day.</Description>
</cards>
<cards id="card_7">
  <card_Cover id="card_playstation"></card_Cover>
  <Name>Playstation Plus Card</Name>
  <card_Type>Digital</card_Type>
```

```

    <card_amount on_stock="yes">$200</card_amount>
    <payment_Method>Paypal, Mobile Banking</payment_Method>
    <validationDuration>Unlimited</validationDuration>
    <shippingCost>$15</shippingCost>
    <shippingAddress>Bristol</shippingAddress>
    <discount_Voucher>35% off from Metro Bank</discount_Voucher>
    <Card_code>5865</Card_code>
    <tax>9%</tax>
    <noOfUsers>600</noOfUsers>
  </cards>
  <cards id="card_8">
    <card_Cover id="card_google"></card_Cover>
    <Name>Google Play Gift Card</Name>
    <card_Type>Physical</card_Type>
    <card_amount on_stock="no">$500</card_amount>
    <payment_Method>Credit Card, Cash, Bill Pay</payment_Method>
    <card_status>Active</card_status>
    <Genre>Music, Movie, Apps</Genre>
    <production_Company>Google Inc.</production_Company>
    <validationDuration>Unlimited</validationDuration>
    <discount_Voucher>40% off from Metro Bank</discount_Voucher>
    <noOfUsers>681</noOfUsers>
    <Description>It can be used to pay for apps, music, and
more.</Description>
  </cards>
  <cards id="card_9">
    <card_Cover id="card_ebay"></card_Cover>
    <Name>eBay Gift Card</Name>
    <card_Type>Digital</card_Type>
    <card_amount on_stock="yes">$50</card_amount>
    <payment_Method>Samsung Pay, Paypal</payment_Method>
    <Genre>Purchasing Items</Genre>
    <validationDuration>6 months</validationDuration>
    <shippingCost>$12</shippingCost>
    <shippingAddress>Newcastle</shippingAddress>
    <discount_Voucher>5% off from Danske Bank</discount_Voucher>
    <Card_code>55665</Card_code>

```

```
<tax>12%</tax>
<noOfUsers>687</noOfUsers>
</cards>
<cards id="card_10">
  <card_Cover id="card_Netflix"></card_Cover>
  <Name>Netflix Gift Card</Name>
  <card_Type>Digital</card_Type>
  <card_amount on_stock="yes">$20</card_amount>
  <payment_Method>Mobile Banking, Paypal, Google Pay</payment_Method>
  <validationDuration>2 months</validationDuration>
  <shippingCost>$11</shippingCost>
  <shippingAddress>Plymouth</shippingAddress>
  <discount_Voucher>12% off from DNB Bank</discount_Voucher>
  <Card_code>6685</Card_code>
  <tax>10%</tax>
  <noOfUsers>600</noOfUsers>
</cards>
<cards id="card_11">
  <card_Cover id="card_roblox"></card_Cover>
  <Name>Roblox Unlimited Card</Name>
  <card_Type>Digital</card_Type>
  <card_amount on_stock="yes">$150</card_amount>
  <payment_Method>Paypal, Google Pay, Samsung Pay</payment_Method>
  <validationDuration>Unlimited</validationDuration>
  <shippingCost>$15</shippingCost>
  <shippingAddress>Edinburgh</shippingAddress>
  <discount_Voucher>55% off from Doha Bank</discount_Voucher>
  <Card_code>8978</Card_code>
  <tax>15%</tax>
  <noOfUsers>125</noOfUsers>
</cards>
<cards id="card_12">
  <card_Cover id="card_shell"></card_Cover>
  <Name>Shell Gift Card</Name>
  <card_Type>Physical</card_Type>
  <card_amount on_stock="no">$120</card_amount>
  <payment_Method>Credit Card, Cash, Bill Pay</payment_Method>
```

```
<card_status>Active</card_status>
<Genre>Gasoline, Auto Supplies</Genre>
<production_Company>Shell Company</production_Company>
<validationDuration>5 months</validationDuration>
<discount_Voucher>10% off from Bank ABC</discount_Voucher>
<noOfUsers>265</noOfUsers>
<Description>It can be used to purchase gasoline and auto supplies at
Shell stations nationwide.</Description>
</cards>
<cards id="card_13">
  <card_Cover id="card_steam"></card_Cover>
  <Name>Steam Gift Card</Name>
  <card_Type>Digital</card_Type>
  <card_amount on_stock="yes">$110</card_amount>
  <payment_Method>Google Pay, Paypal</payment_Method>
  <Genre>Game, Software</Genre>
  <validationDuration>2 months</validationDuration>
  <shippingCost>$10</shippingCost>
  <shippingAddress>Wales</shippingAddress>
  <discount_Voucher>54% off from Lloyds Bank</discount_Voucher>
  <Card_code>5655</Card_code>
  <tax>10%</tax>
  <noOfUsers>656</noOfUsers>
</cards>
<cards id="card_14">
  <card_Cover id="card_valorant"></card_Cover>
  <Name>Valorant</Name>
  <card_Type>Digital</card_Type>
  <card_amount on_stock="no">$258</card_amount>
  <payment_Method>Paypal, samsung Pay</payment_Method>
  <Genre>Gaming</Genre>
  <validationDuration>2 months</validationDuration>
  <shippingCost>$10</shippingCost>
  <shippingAddress>Birmingham</shippingAddress>
  <discount_Voucher>54% off from ING Bank</discount_Voucher>
  <Card_code>000256</Card_code>
  <tax>16%</tax>
```



```

        <noOfUsers>464</noOfUsers>
    </cards>
    <cards id="card_15">
        <card_Cover id="card_Green"></card_Cover>
        <Name>Green Hell PC</Name>
        <card_Type>Digital</card_Type>
        <card_amount on_stock="yes">$500</card_amount>
        <payment_Method>Google Pay, Paypal, Samsung Pay</payment_Method>
        <validationDuration>Unlimited</validationDuration>
        <shippingCost>$15</shippingCost>
        <shippingAddress>Ireland</shippingAddress>
        <discount_Voucher>15% off from Bank Indonesia</discount_Voucher>
        <Card_code>568</Card_code>
        <tax>3%</tax>
        <noOfUsers>159</noOfUsers>
    </cards>
    <cards id="card_16">
        <card_Cover id="card_gentle"></card_Cover>
        <Name>Gentle Herd Gift Card</Name>
        <card_Type>Physical</card_Type>
        <card_amount on_stock="no">$210</card_amount>
        <payment_Method>Credit Card, Cash, Bill Pay</payment_Method>
        <card_status>Active</card_status>
        <Genre>Shopping</Genre>
        <production_Company>Gentle Herd</production_Company>
        <validationDuration>1 year</validationDuration>
        <discount_Voucher>25% off from Lloyds Bank</discount_Voucher>
        <noOfUsers>138</noOfUsers>
        <Description>It gives flexibility to buy whatever your want.</Description>
    </cards>
</giftCards>
<!-- Footer element stores information regarding Copyright-->
<Footer>Steak 44 | Copyright  &#xA9; 2022</Footer>
</cardStore>

```

### 3. Schema Document

```

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="cardStore">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="storeDetails">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="storeName" type="xs:string" />
              <xs:element name="logo">
                <xs:complexType>
                  <xs:attribute name="id" type="xs:string"
use="required" />
                </xs:complexType>
              </xs:element>
              <xs:element name="address" type="xs:string" />
              <xs:element name="contact">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="email" type="xs:string" />
                    <xs:element name="telephone" type="xs:string" />
                    <xs:element name="fax" type="xs:NMTOKEN" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="url" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="giftCards">
          <xs:complexType>
            <xs:sequence>

```

```

<xs:element name="cards" minOccurs="1"
maxOccurs="unbounded">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="card_Cover">
                <xs:complexType>
                    <xs:attribute name="id"
type="xs:ID" use="required" />
                </xs:complexType>
            </xs:element>
            <xs:element name="Name" type="xs:string"
/>
            <xs:element name="card_Type"
type="xs:string" />
            <xs:element name="card_amount"
minOccurs="1" maxOccurs="unbounded">
                <xs:complexType>
                    <xs:simpleContent>
                        <xs:extension
base="xs:string">
                            <xs:attribute
name="on_stock" use="required">
                                <xs:simpleType>
                                    <xs:restriction
base="xs:string">
                                        <xs:enumeratio
n value="no" />
                                        <xs:enumeratio
n value="yes" />
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:attribute>
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>

```

```

        <xs:element name="payment_Method"
type="xs:string" />
        <xs:element name="card_status"
minOccurs="0" maxOccurs="1" type="xs:string" />
        <xs:element name="Genre" minOccurs="0"
maxOccurs="1" type="xs:string" />
        <xs:element name="production_Company"
minOccurs="0" maxOccurs="1" type="xs:string" />
        <xs:element name="validationDuration"
type="xs:string" />
        <xs:element name="shippingCost"
minOccurs="0" maxOccurs="1" type="xs:string" />
        <xs:element name="shippingAddress"
minOccurs="0" maxOccurs="1" type="xs:string" />
        <xs:element name="discount_Voucher"
minOccurs="0" maxOccurs="1" type="xs:string" />
        <xs:element name="Card_code" minOccurs="0"
maxOccurs="1" type="xs:string" />
        <xs:element name="tax" minOccurs="0"
maxOccurs="1" type="xs:string" />
        <xs:element name="noOfUsers"
type="xs:string" />
        <xs:element name="Description"
minOccurs="0" maxOccurs="1" type="xs:string" />
    </xs:sequence>
    <xs:attribute name="id" type="xs:string"
use="required" />
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
    <xs:element name="Footer" type="xs:string" />
</xs:sequence>
    <xs:attribute name="description" type="xs:string" fixed="Gift Card Store
in London" use="required" />
</xs:complexType>

```

```
</xs:element>
</xs:schema>
```

#### 4. Difference between Schema and DTD

The XSD, or XML Schema Definition, is a schema definition document that defines various types of objects and entities in the XML language. Because of its numerous advantages, XSD is widely used for data exchange, particularly in enterprise environments. We'll look at XSD, XSD syntax, and XSD usage in this tutorial (Baydan, 2020).

A set of markup declarations that define a document type for Standard Generalized Markup Language (SGML) languages is referred to as a document type definition (DTD). DTDs use element and attribute-list declarations to define the structure of a class of documents. Parsers use DTD to validate documents. The World Wide Web Consortium has officially recommended it (W3C). DTDs have been largely replaced by XML Namespace-aware schema languages in recent years (techopedia, 2013).

The difference between Schema and DTD are given below:-

XSD or XML Schema	DTD
The structure of an XML document is described by an XML schema. It specifies the elements that can be used in an XML document as well as their attributes, such as whether an element is empty or can hold text. It also specifies which elements are child elements and the order in which they appear.	The structure of documents in SGML-family markup languages such as SGML, XML, and HTML is defined by the DTD. It specifies how document elements are ordered and nested, as well as which elements are included in the documents and their attributes (Indika, 2011).
It is possible to create relationships between elements.	This isn't possible in DTD without causing existing documents to be invalidated.
It is possible to group elements and attributes together to form a single logical unit.	DTD does not allow for the grouping of elements and attributes.
An upper limit for the number of occurrences of an element can be	In DTDs, there is no way to specify an element's upper limit.

specified.	
XSD gives you more control over XML.	XML is less under the control of DTD.
Namespaces and datatypes are supported by XSD.	Namespaces and datatypes are not supported by DTD.
XSD is simple to learn because it does not require the learning of a new language.	Learning DTD is a difficult task.
Inline definitions are not permitted in XML Schema.	Inline definitions are permitted in DTD (Khalid, 2011).
The document structure can be precisely defined in the schema using features like minOccurs and maxOccurs, which estimate the number of times an element can appear in each context.	DTD has limited capabilities in describing document structure in terms of how many elements can nest within other elements due to its format.
The XML schema ensures secure data communication by allowing the sender to describe the data in a way that the receiver can comprehend.	The receiver may misunderstand DTD data.
Schemas provide several benefits for data-intensive workflows.	For text-intensive applications, DTD is preferable.
XSD, on the other hand, has different datatypes depending on the application. String, decimal, float, and Boolean are primitive or built-in datatypes (Atechdaily, 2021).	DTD only supports one data type, String, which is declared in DTD with the #PCDATA keyword.

*Table 3: Difference Between Schema and DTD*

## 5. Testing

The format of XML and Schema (XSD) files is validated in the testing process using an online validator, <http://www.xmlvalidation.com> and it is checked whether XML and CSS files are rendered in the browser.

## 5.1. Test 1

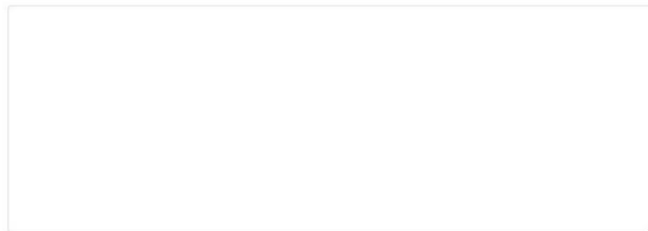
**Objective:** To check if XML and XSD are properly formatted and validated or not.

### Test 1.1

Test No.	1.1
Action	Validation of the XML file by uploading it the validator website.
Expected Result	The validator website to ask for XSD file that was referenced in XML.
Actual Result	The validator website asks for the referenced XSD file.
Conclusion	The test was successful.

*Table 4: Validating XML file*

**Please copy your XML document in here:**



Or upload it:

catalog\_20048874.xml

The validation check is performed against any XML schema or DTD declared inside the XML document.

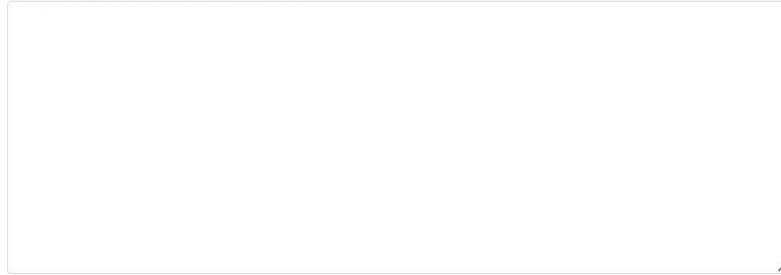
If neither an XML schema nor a DTD is declared, only a syntax check is performed.

To validate the XML document against an external XML schema, click below.

☒ Validate against external XML schema

*Figure 5: Validating the XML file on validator*

Please copy the XML schema in here:



Or upload it:

catalog\_20048874.xsd

The following files have been uploaded so far:

[XML document:](#) 

Click on any file name if you want to edit the file.

*Figure 6: Validating XSD file on validator*

## Test 1.2

Test No.	1.1
Action	Uploading external Schema(XSD) file on the validator website.
Expected Result	The validator website would display “no errors were found” in the formatting of XML and XSD files.
Actual Result	The validator website displays “no errors were found”.
Conclusion	The test was successful.

*Table 5: Validating XML and XSD file with result*



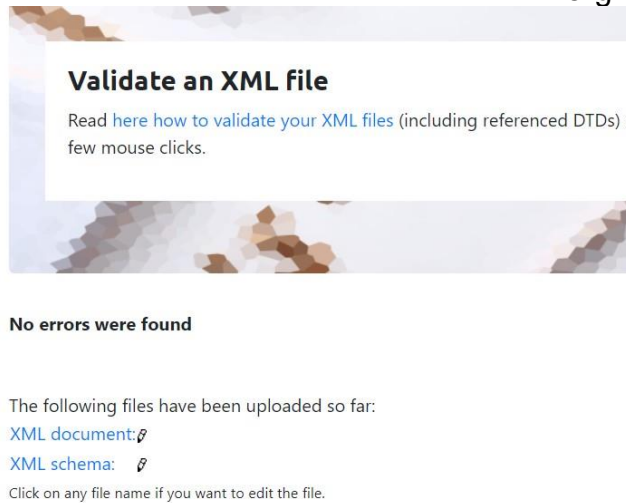


Figure 7: The result of XML and XSD file

## 5.2. Test 2

**Objective: To test whether the optional element in the XML file would work or not.**

Test No.	2
Action	Removing optional element, card_status from the XML file and check whether it will throw an error or not.
Expected Result	After performing the following action an error shouldn't be shown.
Actual Result	The error was not displayed in the text editor as well as browser.
Conclusion	The test was successful.

Table 6: Checking the optional element

```

<cards id="card_2">
  <card_Cover id="card_itunes"></card_Cover>
  <Name>Itunes Gift Card</Name>
  <card_Type>Physical</card_Type>
  <card_amount on_stock="yes">$100</card_amount>
  <payment_Method>Credit Card, Cash, Bill Pay</payment_Method>
  <card_status>Active</card_status>
  <Genre>Music, Movie, Apps</Genre>
  <production_Company>Apple</production_Company>
  <validationDuration>10 months</validationDuration>
  <noOfUsers>554</noOfUsers>
  <Description>iTunes gift card is used to make purchases in the
</cards>

```

Figure 8: Before removing card\_status element

```

edit Selection View Go Run ... catalog_20048874.xml - Visual S...
catalog_20048874.xml X catalog_20048874.xsd # catalog_20048874.css
> ASUS > OneDrive > Desktop > Coursework2 > catalog_20048874.xml > cardStore > giftCards
</cards>
<cards id="card_2">
  <card_Cover id="card_itunes"></card_Cover>
  <Name>Itunes Gift Card</Name>
  <card_Type>Physical</card_Type>
  <card_amount on_stock="yes">$100</card_amount>
  <payment_Method>Credit Card, Cash, Bill Pay</payment_Method>
  <Genre>Music, Movie, Apps</Genre>
  <production_Company>Apple</production_Company>
  <validationDuration>10 months</validationDuration>
  <noOfUsers>554</noOfUsers>
  <Description>iTunes gift card is used to make purchases in the
</cards>

```

Figure 9: After removing card\_status element

### 5.3. Test 3

**Objective:** To test whether the CSS file is working or not.

Test No.	3
Action	XML File is opened in the browser without linking CSS file.
Expected Result	After performing the following action, it shouldn't show the properties of CSS in the browser.
Actual Result	The properties off CSS have not displayed in the browser.
Conclusion	The test was successful.

Table 7: Testing the xml file without using css file

```

1  <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2  <!--
3      Author: Bishal Thapa
4      Date: 03/05/2022
5      Filename: catalog_20048874.xml
6  -->
7
8
9  <!-- cardStore is the root element-->
10 <cardStore description="Gift Card Store in London"
11     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="catalog_20048874.xsd">
12     <!-- cardStore is primarily divided into three child elements storeDetails, giftCards and Footer -->
13     <!-- storeDetails stores storeName, Logo, address, contact and url-->

```

Figure 10: Unlinking the Css file

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```

<!--
  Author: Bishal Thapa
  Date: 03/05/2022
  Filename: catalog_20048874.xml
-->
<!-- cardStore is the root element -->
<cardStore xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" description="Gift Card Store in London" xsi:noNamespaceSchemaLocation="catalog_20048874.xsd">
  <!-- cardStore is primarily divided into three child elements storeDetails, giftCards and Footer -->
  <!-- storeDetails stores storeName, Logo, address, contact and url -->
  <storeDetails>
    <storeName>Steak 44</storeName>
    <!-- Logo of the company is incorporated in above element is referenced in css file -->
    <!-- Also this element contains the address, contact and url -->
    <logo id="Image"/>
    <address>London, United Kingdom</address>
    <!-- There are three child elements inside the contact element. -->
    <contact>
      <email>steak44@gmail.com</email>
      <telephone>(602) 271-4400</telephone>
      <fax> 02-123-5897852</fax>
    </contact>
    <url>uk.steak44.com</url>
  </storeDetails>

```

Figure 11: The output after unlinking CSS file

## 5.4. Test 4

**Objective:** To add the CSS file again and seeing its result in the browser.

Test No.	4
Action	XML File is opened in the browser linking with CSS file.
Expected Result	After performing the following action, it will show the properties of CSS in the browser.
Actual Result	The properties of CSS have been displayed in the browser.
Conclusion	The test was successful.

Table 8: Testing after linking CSS file

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!--
  Author: Bishal Thapa
  Date: 03/05/2022
  Filename: catalog_20048874.xml
-->

<?xml-stylesheet href="catalog_20048874.css"?>
<!-- cardStore is the root element-->
<cardStore description="Gift Card Store in London"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:n
```

Figure 12: Linking the CSS file

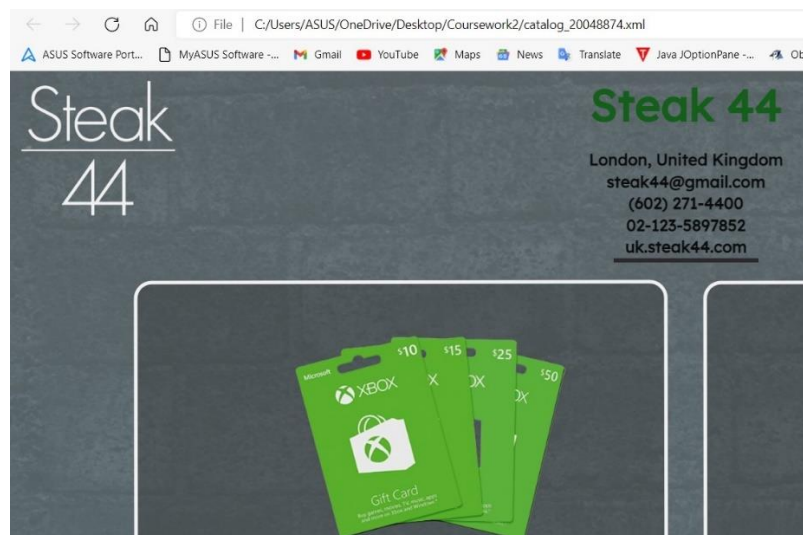


Figure 13: The output after linking the CSS file.

## 5.5. Test 5

**Objective: To check the hover effect that is properly working on CSS or not.**

Test No.	5
Action	At first XML file is opened in the browser and cursor is hovered around picture and text content.
Expected Result	After performing the following action picture should be transform and text content should change its color.
Actual Result	The hover effect has been displayed in the browser.
Conclusion	The test was successful.

Table 9: Testing hover effect



Figure 14: Before hovering the content





Figure 15: After hovering the content.



Figure 16: Hovering the text content.

## 6. Coursework Development

The coursework includes XML, Schema (XSD), and CSS files. A tree diagram was used to design XML.

The following software was used to complete this coursework:

### Visual Studio Code Text Editor(VS Code):

The coursework's contents, such as XML, Schema (XSD), and CSS files, were written by using VS Code Text Editor. The text editor is very easy to use from the beginner to expertise because of its simple interphase. Visual Studio Code (broadly known as VS Code) is a free open-source text editor by Microsoft. Windows, Linux, and macOS are all supported by VS Code. Although the editor is light, it contains some powerful features that have helped VS Code become one of the most popular development environment tools in recent years. VS Code supports a wide range of programming languages, including Java, C++, and Python, as well as CSS, Go, and Docker file. Furthermore, you can add on and even create new extensions for VS Code, such as code linters, debuggers, and cloud and web development support (Mustafeez, 2022). VS Code offers a wide range of services, including a marketplace with many extensions, such as XML for XML support and XML format for simple XML formatting. These extensions made the development process far simpler and more convenient.

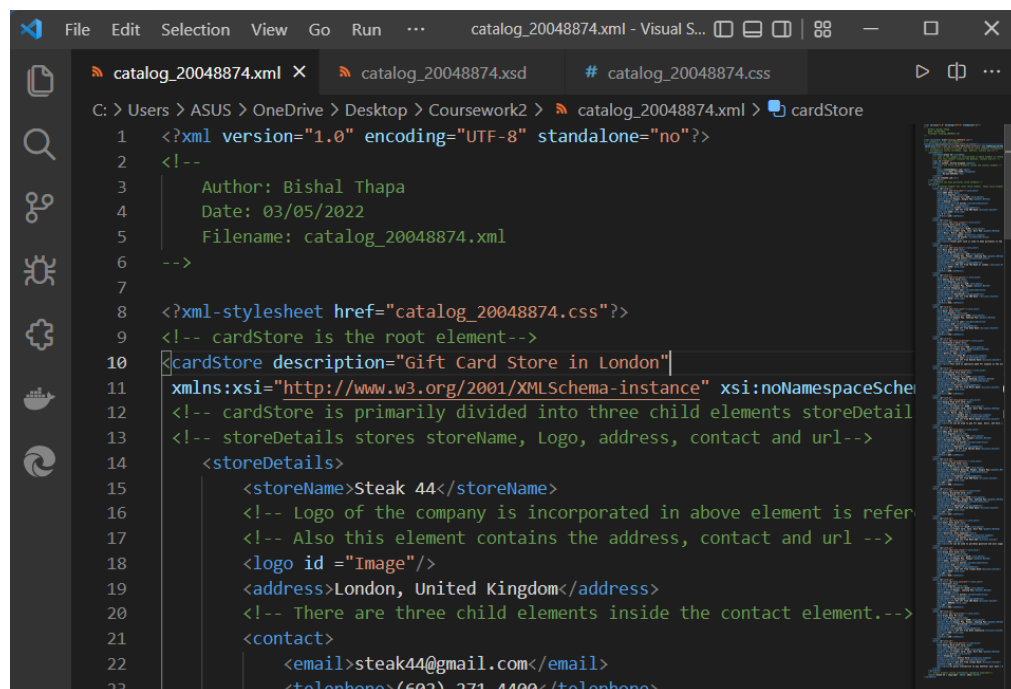


Figure 17: Using VS code text editor for developing the project

**Draw.io:**

Draw.io, created by Seibert Media, is proprietary software for creating diagrams and charts. You can use the software's automatic layout function or create a custom layout. They have a wide range of shapes and hundreds of visual elements to help you create a one-of-a-kind diagram or chart. The drag-and-drop feature makes it easy to create a visually appealing diagram or chart (Computer Hope, 2020). Draw.io was extremely helpful in creating the tree diagram for this project. Its simple user interface made it easy to navigate the website and plot the graph.

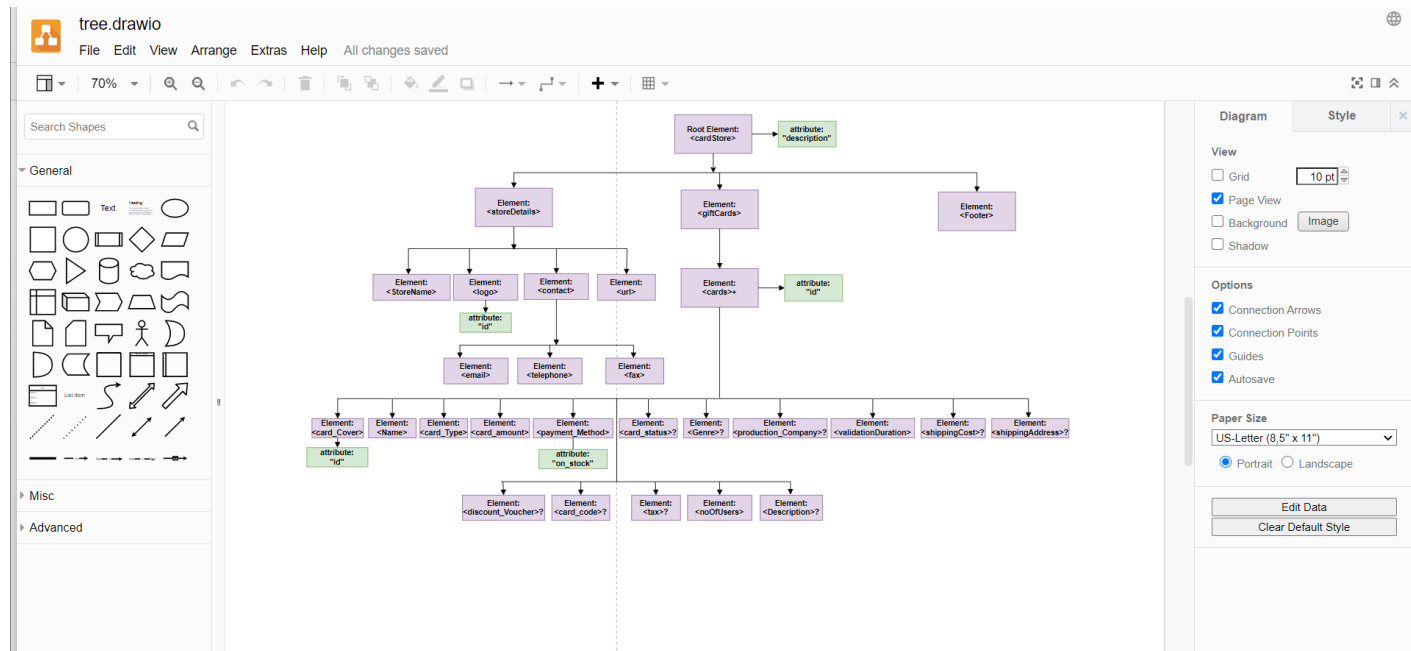


Figure 18: Creating tree diagram by using draw.io

**Online Validation and word development:**

The document part of this coursework was completed by using Microsoft word 2016. Microsoft Word is a word processing program that was first released in 1983 by Microsoft. It is the most widely used word processing program. It is used to create professional-looking documents, letters, reports, resumes, and other documents, as well as to edit or modify existing ones (GeeksforGeeks, 2021).

This XML validation is a website where we can validate our XML and XSD files. This website's URL is <https://www.xmlvalidation.com/>. This website is very useful because it will detect errors in a short amount of time if there are any, and it will also show where those errors are. This website is critical for this coursework's validation.



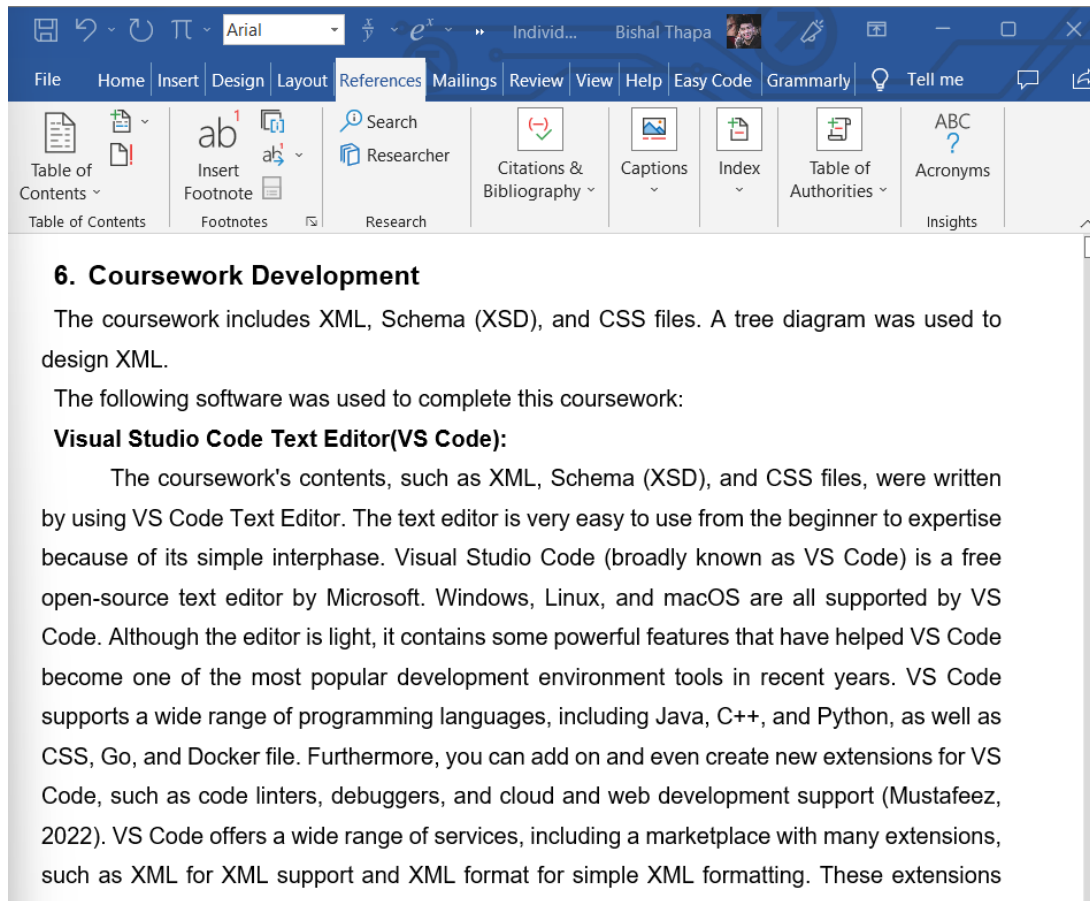


Figure 19: Developing the document file by using MS word

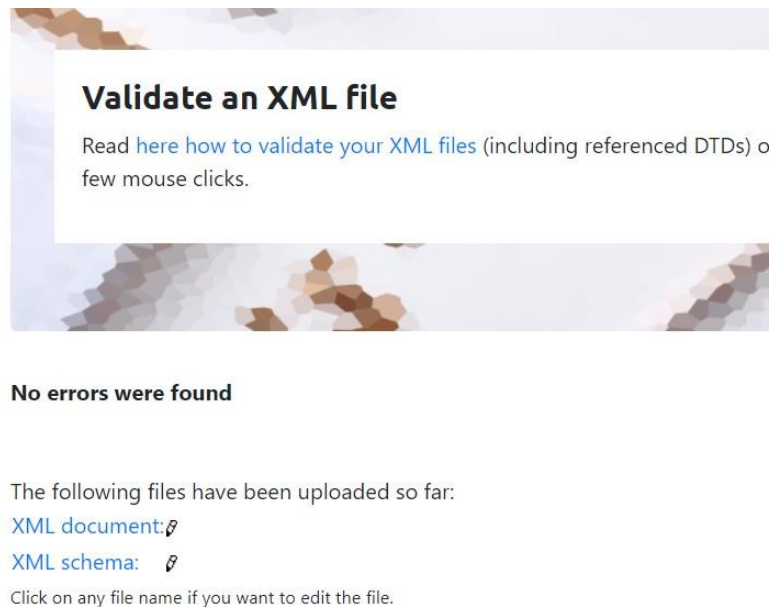


Figure 20: Validating the XML and XSD by using online validator

To design XML file,

A tree diagram was plotted to design an XML file, with all subsequent branches and sub-branches originating from the root element Store, which paved the way for the creation of the XML document by providing it with the necessary structure. The cardStore root element was created first, and then all the child components were added to it. The root element contained the attribute "description." Within the root elements, there were three child components: storeDetails, giftCards, and Footer. The five sub-elements of the storeDetails element were storeName, logo, contact, address, and URL. giftCards has one child element which occurs multiple times. Cards have one attribute "id". Cards contain 16 child elements out of 9 are optional elements. The cards element was a recurring element because it contains the names of the cards, and its child elements contain descriptions of those cards. Only for copyright purposes was the Footer element used. The XML file was created in the context of an XML tree diagram.

To design XSD or Schema file,

In the Schema file, we have used the namespace provided by URI, <http://www.w3.org/2001/XMLSchema>, with the prefix 'xs'. In the XSD, the Russian Doll model has been used. There are eight complex type elements in the coursework. The elements and attributes, as well as their properties, were defined in the schema file. There are various elements in the schema file, such as Simple Elements, Optional Elements, Repeating Elements, and Empty Elements. Optional, required, and fixed were used in the attribute properties. Several data types, such as xs:string and xs:NMTOKEN, were also used in the schema file, depending on the data's use.

## 7. Critical Analysis

Several problems occurred during the project's development because we were unfamiliar with the topic of XML and Schema because it had not been covered in previous courses. The concept of writing an HTML document and CSS, on the other hand, was taught. While working on the coursework, I ran into several issues. I had no problems writing XML files, but I made numerous errors in the schema development process, either carelessly or unintentionally.

Some of the mistakes or issues I ran into while writing the XML file and Schema are listed below.

```

<!-- storeDetails stores storeName, Logo, address, contact and url-->
<storeDetails>
  <storeName>Steak 44</storeName>
  <!-- Logo of the company is incorporated in above element is ref
  <!-- Also this element contains the address, contact and url -->
  <logo id ="Image"/>
  <address>London, United Kingdom</address>
  <!-- There are three child elements inside the contact element.--
  <contact>
    <email>steak44@gmail.com</email>
    <telephone>(602) 271-4400</telephone>
    <fax> 02-123-5897852</fax>
  </contact>
  <url>uk.steak44.com</url>

```

Figure 21: critical analyzing the problem 1

There are several child elements in the <storeDetails> element, as shown in the above diagram. I forgot to close the </storeDetails> element after creating all the child elements, which resulted in an error.

```

<!-- storeDetails stores storeName, Logo, address, contact and url-->
<storeDetails>
  <storeName>Steak 44</storeName>
  <!-- Logo of the company is incorporated in above element is re
  <!-- Also this element contains the address, contact and url --
  <logo id ="Image"/>
  <address>London, United Kingdom</address>
  <!-- There are three child elements inside the contact element.
  <contact>
    <email>steak44@gmail.com</email>
    <telephone>(602) 271-4400</telephone>
    <fax> 02-123-5897852</fax>
  </contact>
  <url>uk.steak44.com</url>
</storeDetails>
<!-- cardStore has many giftCards child elements-->

```

Figure 22: Critical analyzing the solution 1

As illustrated in the preceding figure, I immediately closed the element where the red color or error message has appeared.

```

    <xs:complexType>
      <xs:attribute name="id" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>
  <xs:element name="address" type="xs:string" />
  <xs:element name="contact" type="xs:string">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="email" type="xs:string" />
        <xs:element name="telephone" type="xs:string" />
        <xs:element name="fax" type="xs:NMTOKEN" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

Figure 23: critical analyzing the problem 2

The contact element in the XSD file had child elements, as shown in the above figure, and its data type was mentioned there. However, in the main parent element, I used the data type xs:string by mistake.

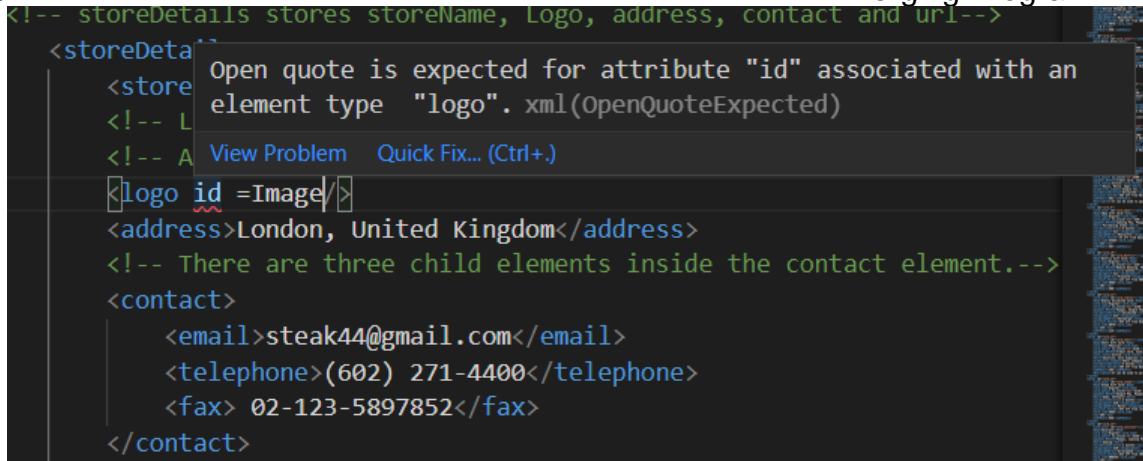
```

<xs:element name="logo">
  <xs:complexType>
    <xs:attribute name="id" type="xs:string" use="required" />
  </xs:complexType>
</xs:element>
<xs:element name="address" type="xs:string" />
<xs:element name="contact">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="email" type="xs:string" />
      <xs:element name="telephone" type="xs:string" />
      <xs:element name="fax" type="xs:NMTOKEN" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

Figure 24: critical analyzing the solution 2

After determining where to use the appropriate data type, I removed the data type xs:string from the parent element, i.e., the contact element, and everything worked fine. The issue was with the XSD file, and it was successfully resolved.



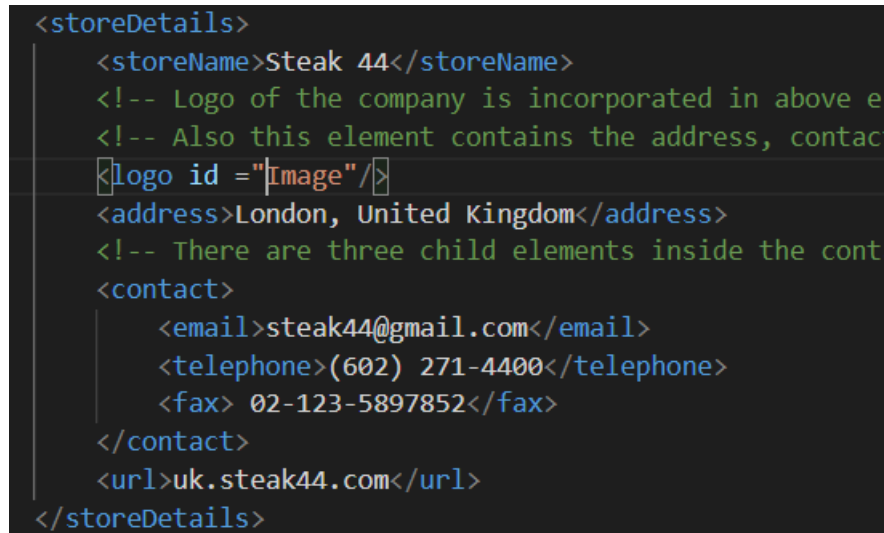
```

<!-- storeDetails stores storeName, Logo, address, contact and url-->
<storeDetails>
  <storeName>Steak 44</storeName>
  <!-- Logo of the company is incorporated in above element -->
  <!-- Also this element contains the address, contact and url -->
  <logo id =Image/>
  <address>London, United Kingdom</address>
  <!-- There are three child elements inside the contact element.-->
  <contact>
    <email>steak44@gmail.com</email>
    <telephone>(602) 271-4400</telephone>
    <fax> 02-123-5897852</fax>
  </contact>
  <url>uk.steak44.com</url>
</storeDetails>

```

Figure 25: critical analyzing the problem 3

As shown in the above figure, I forgot to use a double quote when mentioning the attribute, resulting in the red mark. Furthermore, the error message stated that an open quote is expected for the attribute.



```

<storeDetails>
  <storeName>Steak 44</storeName>
  <!-- Logo of the company is incorporated in above element -->
  <!-- Also this element contains the address, contact and url -->
  <logo id =\"Image\"/>
  <address>London, United Kingdom</address>
  <!-- There are three child elements inside the contact element.-->
  <contact>
    <email>steak44@gmail.com</email>
    <telephone>(602) 271-4400</telephone>
    <fax> 02-123-5897852</fax>
  </contact>
  <url>uk.steak44.com</url>
</storeDetails>

```

Figure 26: critical analyzing the solution 3

After reading the error message on the screen, I realized that a double quote was missing, so I added it to the attribute's value.

```
Name,  
card_Type,  
card_amount,  
payment_Method,  
card_status,  
Genre  
production_Company,  
validationDuration,  
shippingCost,  
shippingAddress,  
discount_Voucher,  
Card_code,  
tax,  
noOfUsers,  
Description {  
  display: block;  
  margin-left: 20%;  
  padding: 0 20px 0 20px;  
  font-weight: bold;
```

Figure 27: Critical analysis of the problem 4

While designing the text content in the CSS section, I accidentally left out a comma, but there was no error displayed.

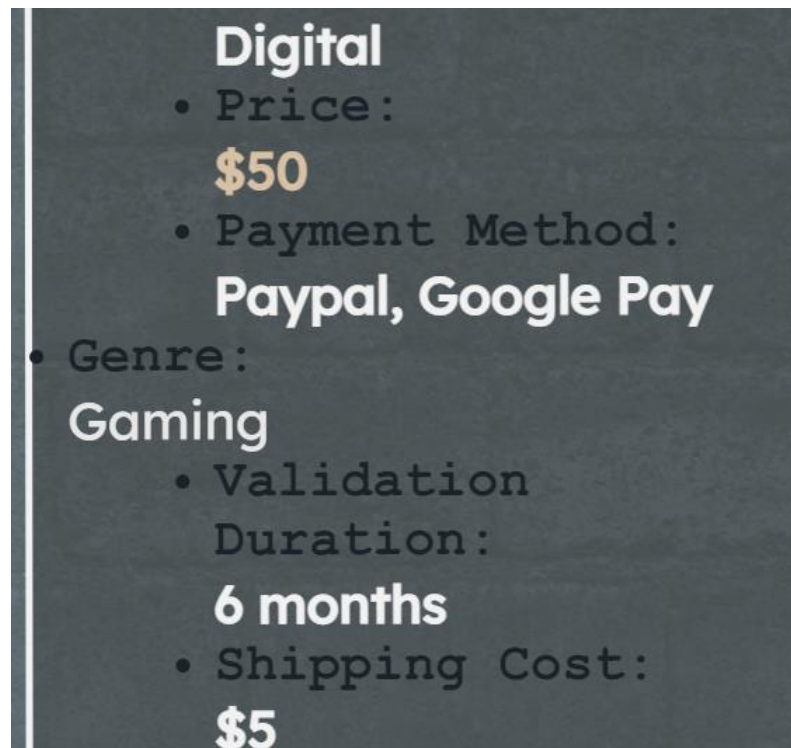


Figure 28: Critical analysis and output of the problem 4

Although there was no error in the text editor, the alignment of the Genre was found to be

incorrect in the web browser because the comma was missed.

```
Name,  
card_Type,  
card_amount,  
payment_Method,  
card_status,  
Genre,  
production_Company,  
validationDuration,  
shippingCost,  
shippingAddress,  
discount_Voucher,  
Card_code,  
tax,  
noOfUsers,  
Description {  
    display: block;
```

Figure 29: Critical analysis of the solution 4

After viewing it in the browser, I noticed that the alignment was unusual and that a comma had been overlooked. So, in CSS, I added a comma after the genre element.

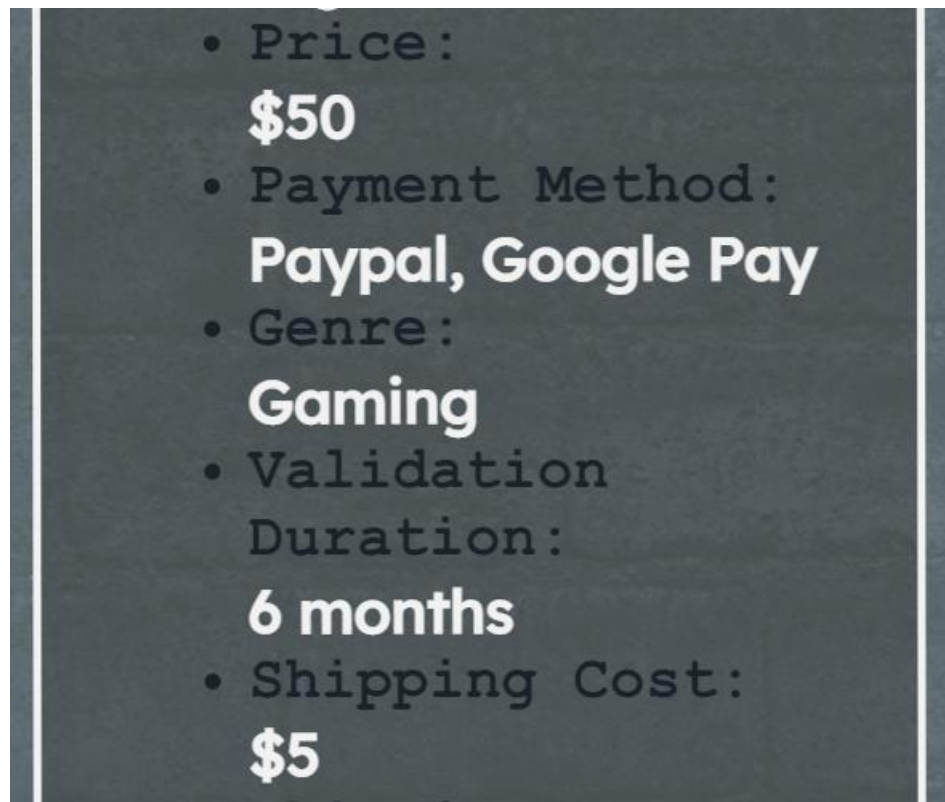


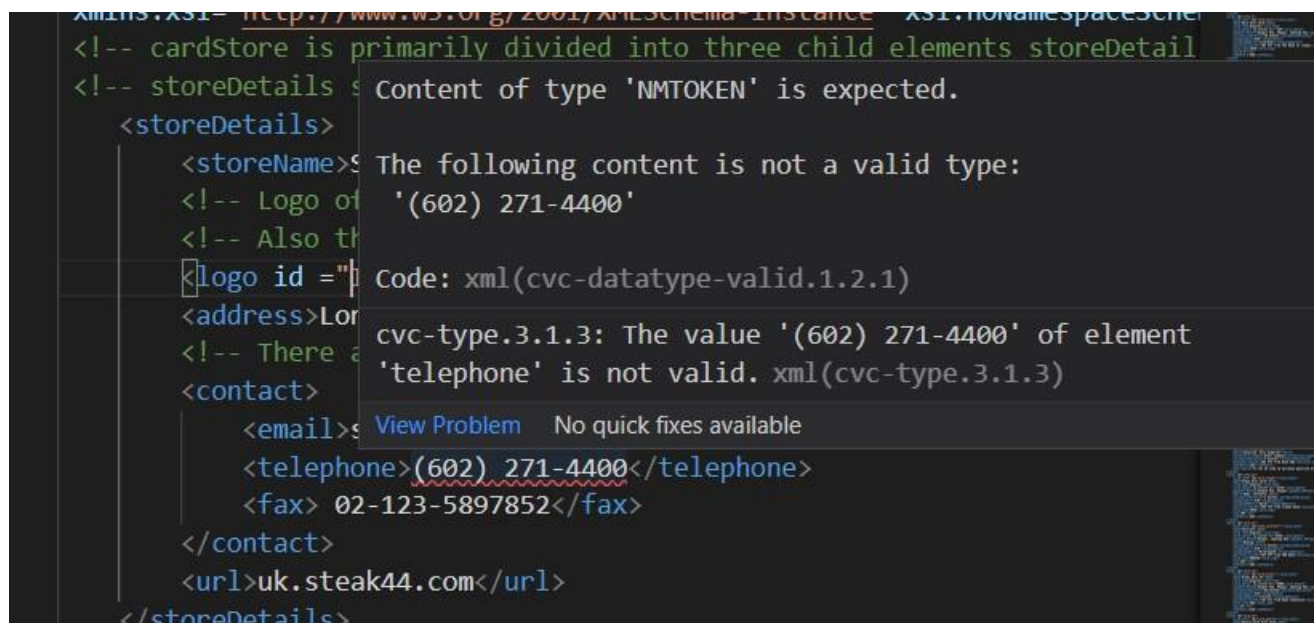
Figure 30: Critical analysis of the solution 4 output



The content and its alignment were properly displayed on the web browser after the comma was added.

```
s:element name="contact">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="email" type="xs:string" />
      <xs:element name="telephone" type="xs:NMTOKEN" />
      <xs:element name="fax" type="xs:NMTOKEN" />
    </xs:sequence>
  </xs:complexType>
</s:element>
s:element name="url" type="xs:string" />
</sequence>
</xs:complexType>
```

Figure 31: Critical analysis of the problem 5



```
<!-- cardStore is primarily divided into three child elements storeDetail
<!-- storeDetails
<storeDetails>
  <storeName>
  <!-- Logo of
  <!-- Also th
  <logo id="
  <address>Lon
  <!-- There a
  <contact>
    <email>
    <telephone>(602) 271-4400</telephone>
    <fax> 02-123-5897852</fax>
  </contact>
  <url>uk.steak44.com</url>
</storeDetails>
```

Content of type 'NMTOKEN' is expected.

The following content is not a valid type:  
'(602) 271-4400'

Code: xml(cvc-datatype-valid.1.2.1)

cvc-type.3.1.3: The value '(602) 271-4400' of element 'telephone' is not valid. xml(cvc-type.3.1.3)

[View Problem](#) No quick fixes available

Figure 32: Critical analysis of the problem output 5

The above figure clearly shows that I used the xs: NMTOKEN data type in the telephone element instead of the xs: string data type. Without the use of a symbol, the phone will not be of the NMTOKEN type. As a result, the issue arises in the XML file.



```

s:element name="contact">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="email" type="xs:string" />
      <xs:element name="telephone" type="xs:string" />
      <xs:element name="fax" type="xs:NMTOKEN" />
    </xs:sequence>
  </xs:complexType>
</s:element>
s:element name="url" type="xs:string" />
sequence>
complexType>

```

Figure 33: Critical analysis of the solution 5

```

<logo id="Image"/>
<address>London, United Kingdom</address>
<!-- There are three child elements inside the contact element.-->
<contact>
  <email>steak44@gmail.com</email>
  <telephone>(602) 271-4400</telephone>
  <fax> 02-123-5897852</fax>
</contact>
<url>uk.steak44.com</url>
</storeDetails>
<!-- cardStore has many giftCards child elements-->
<giftCards>

```

Figure 34: Critical analysis of the solution output 5

The above figure shows that after changing the data type from xs:NMTOKEN to xs:string, the error in the XML file was resolved.

## 8. Conclusion

The assignment is to create a gift card for shop named Steak 44. From the designing stage to the documentation stage, several works have been completed in this coursework. In general, XML, XSD, and CSS were used to create the website. The structure of the document is determined by XML, while schema facilitates the use of various types of data and CSS is used to make the web page appealing.

A Tree diagram was first plotted for the report to give a visual representation of the XML development for the Steak 44 gift card store. The cardStore root element was created first, and then all the child components were added to it such as storeDetails, giftCards and Footer. The root element contained the attribute "description." Basic details such as Store Name, Logo, Address, Telephone Number, and Url are placed in the storeDetails element according to the client's requirements. Card Name, Genre, Price, Validation Duration, Users, including the type with my own added elements, were placed on cards element, and copyright information is found in the footer. To impose a rule on the XML file, an External XSD file was created and linked to the XML file. The coursework contains 29 elements. As stated in the project, an XML Schema was created based on an XML file created using Russian Doll XSD Design. To align the contents of the XML file, CSS properties such as margin, width, and flex were used, as well as properties such as font family, transform, to decorate the website. A website called XML validator was used to validate the XML file and schema. CSS was used to create the various font contents, add the gift card image, hover, and so on. I had a lot of fun doing this coursework, which was mostly in design.

Finally, this coursework has provided me with valuable advice on how to progress as an XML developer. This coursework was a flawless experience in learning professionally and getting hands-on experience in building a system as an XML developer, from gathering the requirements to the end. I struggled with the schema, but with the help of my teacher, I was able to overcome the difficulty. I'd like to express my heartfelt gratitude to my tutor and lecturer for their tireless efforts in assisting me with every problem I've encountered. I'd like to express my gratitude to those who assisted me with my coursework.

## 9. References

- Alexa, G., Apr 11 2008. *WMTIPS*. [Online]  
Available at: <https://www.wmtips.com/css/advantages-using-css/>  
[Accessed 03 05 2022].
- ASQ, 2022. ASQ. [Online]  
Available at: <https://asq.org/quality-resources/tree-diagram#:~:text=A%20tree%20diagram%20is%20a,or%20more%2C%20and%20so%2>  
[Accessed 04 05 2022].
- Atechdaily, 2021. *atechdaily*. [Online]  
Available at: <https://www.atechdaily.com/posts/Difference-between-DTD-and-XSD-schema>  
[Accessed 04 05 2022].
- Baydan, I., 2020. *POFTUT*. [Online]  
Available at: <https://www.poftut.com/what-is-xsd-xml-schema-definition/>  
[Accessed 04 05 2022].
- Computer Hope, 2020. *Computer Hope*. [Online]  
Available at: <https://www.computerhope.com/jargon/d/drawio.htm>  
[Accessed 04 05 2022].
- GeeksforGeeks, 2021. *GeeksforGeeks*. [Online]  
Available at: <https://www.geeksforgeeks.org/introduction-to-microsoft-word/>  
[Accessed 04 05 2022].
- Hemmendinger, D., 2008. *Britannica*. [Online]  
Available at: <https://www.britannica.com/technology/XML>  
[Accessed 03 05 2022].
- I., 2011. *Difference Between.com*. [Online]  
Available at: <https://www.differencebetween.com/difference-between-xml-schema-and-vs-dtd/>  
[Accessed 03 05 2022].
- Khalid, A., 2011. *Difference Between.net*. [Online]  
Available at: <http://www.differencebetween.net/technology/difference-between-xml-schema-and-dtd/>  
[Accessed 04 05 2022].
- Microsoft Build, 2016. XML Schemas (XSD) Starter Kit. *What is XML Schema (XSD)?*, 27 10, pp. 30-34.
- Mustafeez, Z., 2022. *educative*. [Online]  
Available at: <https://www.educative.io/edpresso/what-is-visual-studio-code>  
[Accessed 05 05 2022].
- techopedia, 2013. *techopedia*. [Online]  
Available at: <https://www.techopedia.com/definition/5228/document-type-definition-dtd>  
[Accessed 04 05 2022].
- tutorialspoint, 2022. *tutorialspoint*. [Online]  
Available at: [https://www.tutorialspoint.com/xml/xml\\_overview.htm](https://www.tutorialspoint.com/xml/xml_overview.htm)  
[Accessed 03 05 2022].
- tutorialspoint, 2022. *tutorialspoint*. [Online]  
Available at: [https://www.tutorialspoint.com/css/what\\_is\\_css.htm](https://www.tutorialspoint.com/css/what_is_css.htm)  
[Accessed 04 05 2022].

## 10. Appendix

```
11./*
12.    Author: Bishal Thapa
13.    Date: 03/05/2022
14.    Filename: catalog_20048874.css
15.*/
16./*importing the fonts*/
17.@import
    url('https://fonts.googleapis.com/css2?family=Lexend+Deca&family=Lexend+Zetta&display=swap');
18.
19.cardStore {
20.    background: linear-gradient(rgba(112, 125, 128, 0.734),rgba(107, 253, 241,
    0.525)) , url(./Images/background.jpg) no-repeat fixed center;
21.    background-size: cover;
22.    font-family: 'Lexend Deca',Helvetica, Sans-serif;
23.    filter: contrast(90%);
24.}
25.
26.storeName,
27.address,
28.    email,
29.    telephone,
30.    fax,
31.url {
32.    display: flex;
33.    justify-content: center;
34.    font-size: 18px;
35.    /*One out of four font size*/
36.    color: black;
37.    padding: 5px;
38.    line-height: 0.8;
39.    font-weight: bold;
40.}
41.url {
42.    margin: 0 auto;
43.    width:10%;
44.    border-bottom: 5px solid rgb(36, 31, 36);
45.}
46.
47.storeName {
48.    display: flex;
49.    justify-content: center;
50.    font-family:'Lexend Deca', Verdana, Geneva, Tahoma, sans-serif;
51.    color: rgb(9, 92, 16);;
```

```
52.    /*use of font color in RGB value*/
53.    font-size: 50px;
54.    padding: 30px;
55.    font-weight: bold;
56.}
57.
58.#Image {
59.    background-image: url(Images/steak44.png);
60.    /*displaying logo image*/
61.    width: 200px;
62.    height: 180px;
63.    background-size: 200px;
64.    position: absolute;
65.    top: 1px;
66.}
67.#Image:hover{
68.
69.    transform: scale(0.5);
70.    transition: 1s;
71.    cursor:pointer;
72.    }
73.
74.giftCards {
75.    display: flex;
76.    flex-wrap: wrap;
77.    justify-content: center;
78.}
79.
80.cards {
81.    display: block;
82.    background: rgba(39, 38, 38, 0.35);
83.    border: solid 4px #fff;
84.    border-radius: 20px;
85.    transform: rotate(-0.1deg);
86.    font-size: 30px;
87.    margin: 20px;
88.    width: 550px;
89.    height: 1250px;
90.    padding: 20px;
91.    color: #f4f4f4;
92.    float: left;
93.}
94.
95.card_Cover {
96.    background-size: 380px;
97.    width: 380px;
98.    height: 283px;
```

```
99.     position: absolute;
100.     margin-left: 20%;
101.     margin-top: 20px;
102. }
103.
104. Name,
105. card_Type,
106. card_amount,
107. payment_Method,
108. card_status,
109. Genre,
110. production_Company,
111. validationDuration,
112. shippingCost,
113. shippingAddress,
114. discount_Voucher,
115. Card_code,
116. tax,
117. noOfUsers,
118. Description {
119.     display: block;
120.     margin-left: 20%;
121.     padding: 0 20px 0 20px;
122.     font-weight: bold;
123.     padding-left: 1%;
124.     color: #fff;
125. }
126.
127. Name {
128.     font-family: 'Lexand', Verdana, Geneva, Tahoma, sans-serif;
129.     color: #101820FF;
130.     font-size: 40px;
131.     padding: 10px;
132.     font-weight: bold;
133.     margin-top: 350px;
134.     color: rgb(123, 165, 17);
135.     margin-left: 25%;
136.
137. }
138. Name::before,
139. card_Type::before,
140. card_amount::before,
141. payment_Method::before,
142. card_status::before,
143. Genre::before,
144. production_Company::before,
```

```
145. validationDuration::before,
146. shippingCost::before,
147. shippingAddress::before,
148. discount_Voucher::before,
149. Card_code::before,
150. tax::before,
151. noOfUsers::before,
152. Description::before
153. {
154.     font-weight: bold;
155.     color: #101820FF;
156.     font-family: 'Courier New', Courier, monospace;
157.     display: list-item;
158.     list-style-type: disc;
159. }
160. /*hovering effect in the text */
161. card_Type: hover,
162. card_amount: hover,
163. payment_Method: hover,
164. card_status: hover,
165. Genre: hover,
166. production_Company: hover,
167. validationDuration: hover,
168. shippingCost: hover,
169. shippingAddress: hover,
170. discount_Voucher: hover,
171. Card_code: hover,
172. tax: hover,
173. noOfUsers: hover,
174. Description: hover {
175.     color: rgb(226, 200, 168);
176.     cursor: pointer;
177. }
178. /*list-item is used in order to create a list */
179.
180. Name::before {
181.     content: "Card Name: ";
182. }
183.
184. card_Type::before {
185.     content: "Type: ";
186. }
187.
188. card_amount::before {
189.     content: "Price: ";
190. }
191. payment_Method::before {
```

```
192.     content: "Payment Method: ";
193. }
194. card_status::before {
195.     content: "Card Status: ";
196. }
197. Genre::before {
198.     content: "Genre: ";
199. }
200. production_Company::before {
201.     content: "Production Company: ";
202. }
203.
204. validationDuration::before {
205.     content: "Validation Duration: ";
206. }
207.
208. shippingCost::before {
209.     content: "Shipping Cost: ";
210. }
211.
212. shippingAddress::before {
213.     content: "Shipping Address: ";
214. }
215. discount_Voucher::before {
216.     content: "Discount Voucher: ";
217. }
218.
219. Card_code::before {
220.     content: "Code: "
221. }
222.
223. tax::before {
224.     content: "Tax: "
225. }
226.
227. noOfUsers::before {
228.     content: "Total Users: "
229. }
230. Description::before{
231.     content: "Description: "
232. }
233. /*displaying image of each gift cards */
234. #card_xbox {
235.     background-image: url(./Images/Xbox.png);
236. }
237.
238. #card_itunes{
```



```
239.     background-image: url(./Images/Itunes.png);
240. }
241.
242. #card_hulu {
243.     background-image: url(./Images/Hulu.png);
244. }
245.
246. #card_amazon {
247.     background-image: url(./Images/Amazon.png);
248. }
249.
250. #card_razer {
251.     background-image: url(./Images/Razer.png);
252. }
253.
254. #card_valentine{
255.     background-image: url(./Images/valentine.png);
256. }
257.
258. #card_playstation {
259.     background-image: url(./Images/playstation.png);
260. }
261.
262. #card_google {
263.     background-image: url(./Images/google.png);
264. }
265.
266. #card_ebay {
267.     background-image: url(./Images/eBay.png);
268. }
269.
270. #card_Netflix{
271.     background-image: url(./Images/Netflix.png);
272. }
273.
274. #card_roblox {
275.     background-image: url(./Images/Roblox.png);
276. }
277.
278. #card_shell {
279.     background-image: url(./Images/Shell.png);
280. }
281.
282. #card_steam {
283.     background-image: url(./Images/steam.png);
284. }
285.
```

```
286. #card_valorant {
287.     background-image: url(./Images/valorant.png);
288. }
289.
290. #card_Green {
291.     background-image: url(./Images/greenhell.png);
292. }
293.
294. #card_gentle {
295.     background-image: url(./Images/gentleherd.png);
296. }
297. /*putting hover in gift cards images*/
298. #card_xbox:hover,#card_itunes:hover,#card_hulu:hover,#card_amazon:hover,#card_ra
    zer:hover,#card_valentine:hover,#card_playstation:hover,
299. #card_google:hover,#card_ebay:hover,#card_Netflix:hover,#card_roblox:hover,#card
    _shell:hover,#card_steam:hover,#card_valorant:hover,
300. #card_Green:hover,#card_gentle:hover{
301.     transform: scale(1.3);
302.     transition: 0.8s;
303.     cursor:pointer;
304. }
305. /*adjusting the image for gift cards*/
306. #card_shell {
307.     background-size: 400px;
308.     width: 380px;
309.     height: 230px;
310.     position: absolute;
311.     margin-left: 20%;
312.     margin-top: 20px;
313. }
314. #card_valorant{
315.     background-size: 350px;
316.     width: 360px;
317.     height: 310px;
318.     position: absolute;
319.     margin-left: 20%;
320.     margin-top: 20px;
321. }
322. #card_steam{
323.     background-size: 400px;
324.     width: 380px;
325.     height: 350px;
326.     position: absolute;
327.     margin-left: 20%;
328.     margin-top: 20px;
329. }
330. #card_valentine{
```

```
331.     background-size: 320px;
332.     width: 310px;
333.     height: 280px;
334.     position: absolute;
335.     margin-left: 20%;
336.     margin-top: 20px;
337. }
338. #card_roblox{
339.     background-size: 350px;
340.     width: 350px;
341.     height: 310px;
342.     position: absolute;
343.     margin-left: 20%;
344.     margin-top: 20px;
345. }
346. #card_amazon{
347.     background-size: 363px;
348.     width: 360px;
349.     height: 280px;
350.     position: absolute;
351.     margin-left: 20%;
352.     margin-top: 20px;
353. }
354. #card_Green{
355.     background-size: 340px;
356.     width: 340px;
357.     height: 335px;
358.     position: absolute;
359.     margin-left: 20%;
360.     margin-top: 20px;
361. }
362. #card_gentle{
363.     background-size: 348px;
364.     width: 340px;
365.     height: 301px;
366.     position: absolute;
367.     margin-left: 20%;
368.     margin-top: 20px;
369. }
370.
371. Footer{
372.     padding: 20px;
373.     margin-top: 20px;
374.     background-color: black;
375.     color: #f4f4f4;
376.     display: flex;
377.     justify-content: center;
```

```
378. }
```