



**Module Code & Module Title**

**CS5004NT Emerging Programming Platforms and Technologies**

**Assessment Weightage & Type**

**30% Individual Coursework**

**Year and Semester**

**2021 Spring**

**Student Name: Susan Shrestha**

**London Met ID: 19033540**

**College ID: NP05CP4S200039**

**Assignment Due Date: 7 May, 2021**

**Assignment Submission Date: 7 May, 2021**

*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 mark of zero will be awarded.*

## Table of Contents

INTRODUCTION.....	1
TREE DIAGRAM.....	2
DEVELOPMENT .....	3
XML content .....	3
DTD content .....	11
SCHEMA (XSD content).....	12
CSS content .....	16
XML SCHEMA VS DTD .....	22
TESTING.....	23
Test 1: WELL-FORMED .....	23
Test 2: VALID (DTD).....	26
Test 3: VALID (XSD).....	28
Test 4: VALID CSS LINKING.....	30
Test 5: VALIDATING ENUMERATION .....	32
Test 6: VALIDATING PATTERNED VALUES.....	34
Test 7: REQUIRED ATTRIBUTE VALUE .....	37
Test 8: MATCHING TAG .....	40
DEVELOPMENT PROCESS.....	43
CRITICAL ANALYSIS AND CONCLUSION .....	45
REFERENCES.....	47

## Table of Figures

Figure 1: XML Tree Diagram of Music Store .....	2
Figure 2: Well-formed XML Code .....	24
Figure 3: Well-formed XML document loaded on browser .....	25
Figure 4: Uploading XML code for validation .....	26
Figure 5: Uploading DTD file for validation .....	27
Figure 6: Successfully validated result .....	27
Figure 7: Uploading XML file for validation .....	28
Figure 8: Uploading XSD (Schema) file for validation .....	29
Figure 9: Successfully validated XML Schema result .....	29
Figure 10: Linking CSS file to XML document .....	30
Figure 11: CSS linked XML document .....	31
Figure 12: CSS linked XML document .....	31
Figure 13: Changing enumerated attribute values .....	32
Figure 14: Uploading XML Code for enumeration validation .....	33
Figure 15: Uploading Schema file for enumeration validation .....	33
Figure 16: Result of enumeration validation .....	34
Figure 17: Changing patterned value for validation .....	35
Figure 18: Uploading XML code for patterned value validation .....	36
Figure 19: Uploading Schema file for patterned value validation .....	36
Figure 20: Result of patterned value validation .....	37
Figure 21: Removing required type attribute .....	38
Figure 22: Uploading XML code .....	38
Figure 23: Uploading Schema file for validating XML .....	39
Figure 24: Result of required attribute validation .....	39
Figure 25: Removing an ending tag from XML code .....	40
Figure 26: Uploading XML document for matching tag validation .....	41
Figure 27: Uploading Schema for matching tag validation .....	41
Figure 28: Result of matching tag validation .....	42

## Table of Tables

Table 1: Schema vs DTD .....	23
Table 2: Test case: well-formed XML document .....	23
Table 3: Test Case: Valid (DTD) .....	26
Table 4: Test Case: Valid (XSD) .....	28
Table 5: Test Case: Valid CSS linking.....	30
Table 6: Test Case: Enumeration Validation .....	32
Table 7: Test Case: Patterned values validation .....	34
Table 8: Test Case: Required Attribute Value validation .....	37
Table 9: Test Case: Matching Tag Validation.....	40

## INTRODUCTION

To commence with, this particular report is on the second coursework of “Emerging Programming Platforms and Technologies” module where the work is to be done starting with modelling a system for a music store as per the given scenario. The name for music store is decided as “Divine Music Store”. The scenario provides specifications about store details and contents to be included in the XML document such as song title, genre, singer, producer, director, lyricist, release year and so on. From these specifications, the elements of the XML document are to be analyzed for preparing tree structure for the efficient and ease while writing the XML document, DTD and XML schema. Next task is to create a CSS file to render the XML data on web browser. For the verification and validation of well-formed XML, testing will be carried out.

Extensible Markup Language (**XML**) is a very flexible text format inherited from Standard Generalized Markup Language (**SGML**) that is originally designed to address the problem of large-scale electronic publication, but it is now particularly relevant in the sharing of broad range of data on the web and everywhere else. (liam, 2016) XML allows users to create own self-descriptive tags that fits the application and store the data irrespective of how it will be presented. XML was developed by World Wide Web Consortium (**W3C**). XML can be used to exchange the data between organizations and systems, store and arrange the data that can customize data handling needs, merge with stylesheets to create any desired output. (tutorialspoint, n.d.)

A Document Type Definition (**DTD**) is a set of markup declarations that define and constraints a document type for XML or SGML documents. It defines the structure of class of documents through element and attribute declarations and helps parsers validate documents. DTD is officially recommended by W3C. (techopedia, n.d.)

An XML Schema Definition (**XSD**) describes the structure of an XML document. It defines the legal building blocks of an XML document such as elements and attributes and its data types, number and order of child elements, default and fixed values for elements and attributes. XSD is an alternative to DTD as XSD supports data types, facets and patterns that supports correctness of data. (w3schools, n.d.)

## TREE DIAGRAM

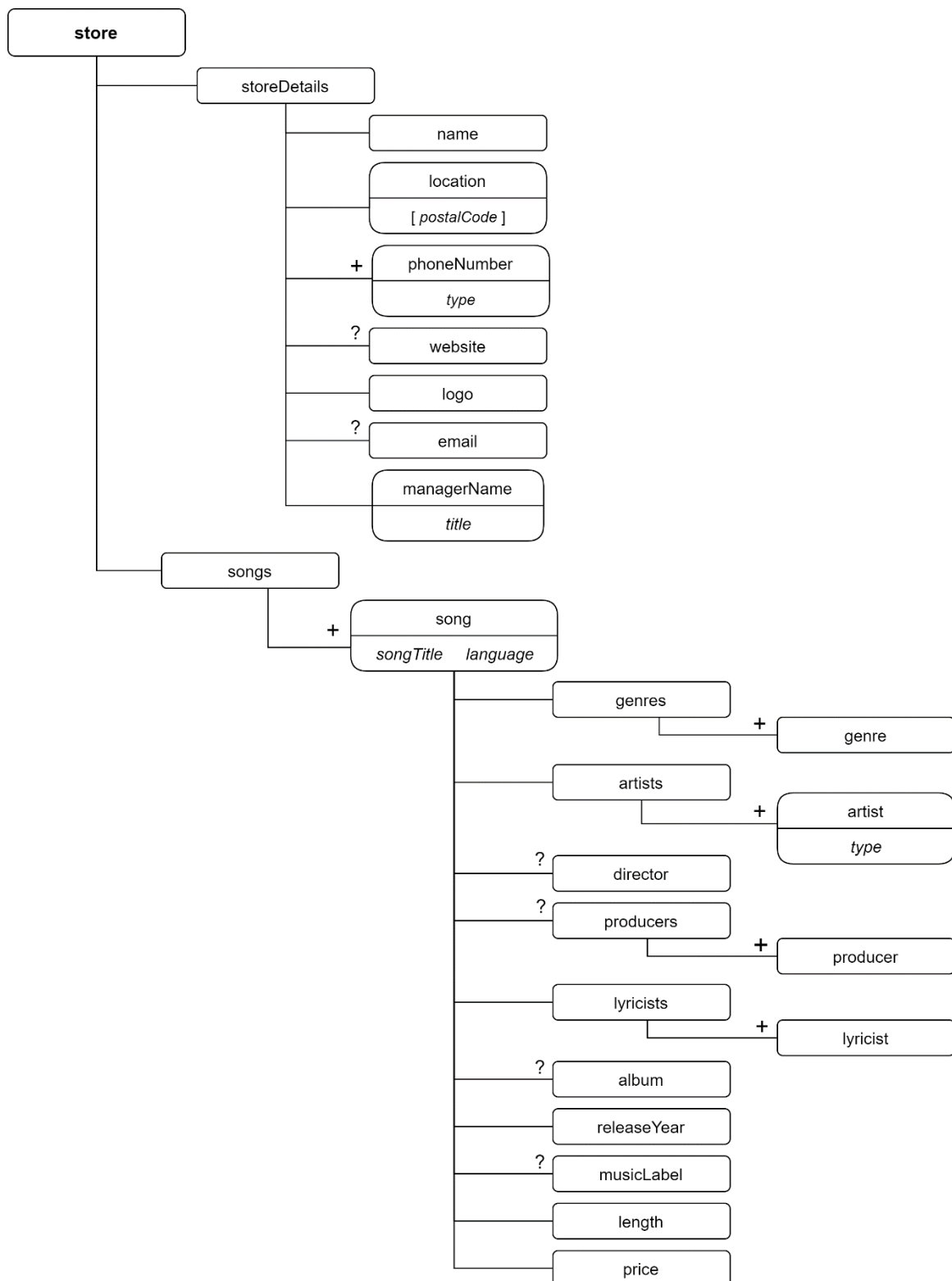


Figure 1: XML Tree Diagram of Music Store

## DEVELOPMENT

### XML content

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
<!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd" -->
<store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
<!-- <store> -->
  <storeDetail>
    <name>Divine Music Store</name>
    <location postalCode="56600">Salakpur, Morang</location>
    <phoneNumber type="Telephone">021545720</phoneNumber>
    <phoneNumber type="Mobile">9819090917</phoneNumber>
    <website>www.divinemusicstore.com.np</website>
    <logo/>
    <email>divinemusicstore@gmail.com</email>
    <managerName title="Mr.">Susan Shrestha</managerName>
  </storeDetail>

  <songs>
    <song songTitle="Sweet child O' mine" language="English">
      <genres>
        <genre>Hard Rock</genre>
        <genre>Glam Metal</genre>
      </genres>
      <artists>
        <artist type="Band">Guns N' Roses</artist>
      </artists>
      <director>Nigel Dick</director>
      <producers>
        <producer>Mike Clink</producer>
      </producers>
      <lyricists>
        <lyricist>Guns N' Roses</lyricist>
      </lyricists>
      <album>Appetite for Destruction</album>
      <releaseYear>1988</releaseYear>
      <musicLabel>Geffen</musicLabel>
      <length>05:02</length>
      <price>Rs.900</price>
    </song>

    <song songTitle="लेकाली चोयाको डोको" language="Nepali">
```

```
<genres>
  <genre>Folk</genre>
</genres>
<artists>
  <artist type="Singer">Ram Thapa</artist>
</artists>
<director>Sudan Sharma</director>
<producers>
  <producer>Music Nepal</producer>
</producers>
<lyricists>
  <lyricist>Sudan Sharma</lyricist>
</lyricists>
<album>Thokna Madal Thok</album>
<releaseYear>2000</releaseYear>
<musicLabel>Music Nepal</musicLabel>
<length>02:55</length>
<price>Rs.800</price>
</song>

<song songTitle="I Want to Break Free" language="English">
  <genres>
    <genre>Rock</genre>
    <genre>Synth-Pop</genre>
    <genre>Classic Rock</genre>
  </genres>
  <artists>
    <artist type="Band">Queen</artist>
  </artists>
  <producers>
    <producer>Queen</producer>
    <producer>Reinhold Mack</producer>
  </producers>
  <lyricists>
    <lyricist>John Deacon</lyricist>
  </lyricists>
  <album>The Works</album>
  <releaseYear>1984</releaseYear>
  <musicLabel>EMI(UK)</musicLabel>
  <length>03:19</length>
  <price>Rs.2000</price>
</song>

<song songTitle="Galbandi" language="Nepali">
  <genres>
```



```
<genre>Folk</genre>
</genres>
<artists>
  <artist type="Singer">Prakash Saput</artist>
  <artist type="Singer">Shanti Shree Pariyar</artist>
</artists>
<director>Prakash Saput</director>
<lyricists>
  <lyricist>Prakash Saput</lyricist>
</lyricists>
<album>Bola Maya</album>
<releaseYear>2019</releaseYear>
<musicLabel>Bhaka Nepal Pvt. Ltd.</musicLabel>
<length>15:34</length>
<price>Rs.300</price>
</song>

<song songTitle="K Pais Nepali?" language="Nepali">
  <genres>
    <genre>Rock</genre>
  </genres>
  <artists>
    <artist type="Band">The Shadows 'Nepal'</artist>
  </artists>
  <director>Rohit Shakya</director>
  <producers>
    <producer>Rohit Shakya</producer>
  </producers>
  <lyricists>
    <lyricist>Swapnil Sharma</lyricist>
    <lyricist>Amit Pradhan</lyricist>
  </lyricists>
  <releaseYear>2019</releaseYear>
  <musicLabel>TuneCore</musicLabel>
  <length>05:13</length>
  <price>Rs.1000</price>
</song>

<song songTitle="All of Me" language="English">
  <genres>
    <genre>R&B</genre>
    <genre>Pop</genre>
    <genre>Soul</genre>
  </genres>
  <artists>
```

```
        <artist type="Singer">John Legend</artist>
    </artists>
    <director>NABIL</director>
    <producers>
        <producer>Dave Tozer</producer>
        <producer>John Legend</producer>
    </producers>
    <lyricists>
        <lyricist>Toby Gad</lyricist>
        <lyricist>John Legend</lyricist>
    </lyricists>
    <album>Alto Astral - Internacionales</album>
    <releaseYear>2013</releaseYear>
    <musicLabel>GOOD</musicLabel>
    <length>04:30</length>
    <price>Rs.500</price>
</song>

<song songTitle="Something" language="English">
    <genres>
        <genre>Rock</genre>
        <genre>Pop</genre>
    </genres>
    <artists>
        <artist type="Band">The Beatles</artist>
    </artists>
    <producers>
        <producer>George Martin</producer>
    </producers>
    <lyricists>
        <lyricist>George Harrison</lyricist>
    </lyricists>
    <album>Abbey Road</album>
    <releaseYear>1969</releaseYear>
    <musicLabel>Apple</musicLabel>
    <length>02:59</length>
    <price>Rs.1500</price>
</song>

<song songTitle="Despacito" language="Spanish">
    <genres>
        <genre>Latin Pop</genre>
        <genre>Reggaeton</genre>
    </genres>
    <artists>
```

```
<artist type="Singer">Luis Fonsi</artist>
<artist type="Rapper">Daddy Yankee</artist>
</artists>
<producers>
  <producer>Mauricio Rengifo</producer>
  <producer>Andrés Torres</producer>
</producers>
<lyricists>
  <lyricist>Lui Rodriguez</lyricist>
  <lyricist>Erika Ender</lyricist>
  <lyricist>Ramon Ayala</lyricist>
</lyricists>
<album>Vida</album>
<releaseYear>2017</releaseYear>
<musicLabel>Universal Latin</musicLabel>
<length>03:47</length>
<price>Rs.600</price>
</song>

<song songTitle="Waka Waka(This Time for Africa)" language="Spanish">
  <genres>
    <genre>Afro Fusion</genre>
  </genres>
  <artists>
    <artist type="Singer">Shakira</artist>
  </artists>
  <director>Marcus Raboy</director>
  <producers>
    <producer>John Hill</producer>
  </producers>
  <lyricists>
    <lyricist>John Hill</lyricist>
    <lyricist>Shakira</lyricist>
  </lyricists>
  <album>Sale el Sol</album>
  <releaseYear>2010</releaseYear>
  <musicLabel>Epic</musicLabel>
  <length>03:22</length>
  <price>Rs.400</price>
</song>

<song songTitle="Tu Hai" language="Hindi">
  <genres>
    <genre>Bollywood</genre>
  </genres>
```

```
<artists>
  <artist type="Singer">A.R. RAHMAN</artist>
  <artist type="Singer">SANAH MOIDUTTY</artist>
</artists>
<director>A.R. RAHMAN</director>
<producers>
  <producer>T-Series</producer>
</producers>
<lyricists>
  <lyricist>JAVED AKHTAR</lyricist>
</lyricists>
<album>MOHENJO DARO</album>
<releaseYear>2016</releaseYear>
<length>03:58</length>
<price>Rs.100</price>
</song>

<song songTitle="Take Me Home, Country Roads" language="English">
  <genres>
    <genre>Country</genre>
  </genres>
  <artists>
    <artist type="Singer">John Denver</artist>
  </artists>
  <producers>
    <producer>Milton Okun</producer>
    <producer>Susan Ruskin</producer>
  </producers>
  <lyricists>
    <lyricist>Bill Danoff</lyricist>
    <lyricist>Taffy Nivert</lyricist>
    <lyricist>John Denver</lyricist>
  </lyricists>
  <album>Poems, Prayers and Promises</album>
  <releaseYear>1971</releaseYear>
  <musicLabel>RCA</musicLabel>
  <length>03:17</length>
  <price>Rs.1600</price>
</song>

<song songTitle="Hit the Road Jack" language="English">
  <genres>
    <genre>Jazz</genre>
    <genre>R&B</genre>
  </genres>
  <artists>
    <artist type="Singer">Ray Charles</artist>
  </artists>
  <producers>
    <producer>Tommy Raggs</producer>
  </producers>
  <lyricists>
    <lyricist>Tommy Raggs</lyricist>
  </lyricists>
  <album>The Soul of Tommy Raggs</album>
  <releaseYear>1961</releaseYear>
  <musicLabel>Mercury</musicLabel>
  <length>03:11</length>
  <price>Rs.1200</price>
</song>
```

```
</genres>
<artists>
  <artist type="Singer">Ray Charles</artist>
</artists>
<producers>
  <producer>Sid Feller</producer>
</producers>
<lyricists>
  <lyricist>Percy Mayfield</lyricist>
</lyricists>
<album>Ray Charles Greatest Hits</album>
<releaseYear>1962</releaseYear>
<musicLabel>ABC-Paramount</musicLabel>
<length>02:00</length>
<price>Rs.400</price>
</song>

<song songTitle="Kathmandu Bazar" language="Nepali">
  <genres>
    <genre>Pop</genre>
  </genres>
  <artists>
    <artist type="Singer">Uday Raj Poudel</artist>
  </artists>
  <director>Sabin Karki</director>
  <lyricists>
    <lyricist>Sabin Ektaare</lyricist>
  </lyricists>
  <releaseYear>2021</releaseYear>
  <length>05:48</length>
  <price>Rs.1100</price>
</song>

<song songTitle="24K Magic" language="English">
  <genres>
    <genre>Funk</genre>
    <genre>disco</genre>
    <genre>R&B</genre>
  </genres>
  <artists>
    <artist type="Singer">Bruno Mars</artist>
  </artists>
  <producers>
    <producer>Shampoo Press & Curl</producer>
    <producer>The Stereotypes</producer>
```

```
</producers>
<lyricists>
  <lyricist>Bruno Mars</lyricist>
  <lyricist>Philip Lawrence</lyricist>
  <lyricist>Christopher Brody Brown</lyricist>
</lyricists>
<album>24K Magic</album>
<releaseYear>2016</releaseYear>
<musicLabel>Atlantic</musicLabel>
<length>03:46</length>
<price>Rs.900</price>
</song>

<song songTitle="Mast Magan" language="Hindi">
  <genres>
    <genre>Soul</genre>
  </genres>
  <artists>
    <artist type="Singer">Arijit Singh</artist>
    <artist type="Singer">Chinmayi Sripada</artist>
  </artists>
  <director>Shankar Eshaan Loy</director>
  <producers>
    <producer>T-Series</producer>
  </producers>
  <lyricists>
    <lyricist>Amitabh Bhattacharya</lyricist>
  </lyricists>
  <album>2 States</album>
  <releaseYear>2014</releaseYear>
  <musicLabel>T-Series</musicLabel>
  <length>03:43</length>
  <price>Rs.400</price>
</song>
</songs>
</store>
```

## DTD content

```
<?xml version="1.0" encoding="utf-8"?>

<!ELEMENT store (storeDetail,songs)>

<!ELEMENT storeDetail (name,location,phoneNumber+,website?,logo,email?,managerName)>

<!ELEMENT songs (song)+>

<!ELEMENT name (#PCDATA)>

<!ELEMENT location (#PCDATA)>
<ATTLIST location postalCode CDATA #IMPLIED>

<!ELEMENT phoneNumber (#PCDATA)>
<ATTLIST phoneNumber type (Telephone|Mobile) #REQUIRED>

<!ELEMENT website (#PCDATA)>

<!ELEMENT logo EMPTY>

<!ELEMENT email (#PCDATA)>

<!ELEMENT managerName (#PCDATA)>
<ATTLIST managerName title (Mr. | Mrs. | Ms.) #REQUIRED>

<!ELEMENT song (genres,artists,director?,producers?,lyricists,album?,releaseYear,musicLabel?,
length,price)>
<ATTLIST song
    language NMTOKEN #REQUIRED
    songTitle CDATA #REQUIRED>

<!ELEMENT genres (genre)+>

<!ELEMENT artists (artist)+>

<!ELEMENT director (#PCDATA)>

<!ELEMENT producers (producer)+>

<!ELEMENT lyricists (lyricist)+>

<!ELEMENT album (#PCDATA)>
```

```

<!ELEMENT releaseYear (#PCDATA)>

<!ELEMENT musicLabel (#PCDATA)>

<!ELEMENT length (#PCDATA)>

<!ELEMENT price (#PCDATA)>

<!ELEMENT genre (#PCDATA)>

<!ELEMENT artist (#PCDATA)>
<!ATTLIST artist type NMTOKEN #REQUIRED>

<!ELEMENT producer (#PCDATA)>

<!ELEMENT lyricist (#PCDATA)>

```

## SCHEMA (XSD content)

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="store">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="storeDetail">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="name" type="xs:string" />
              <xs:element name="location">
                <xs:complexType>
                  <xs:simpleContent>
                    <xs:extension base="xs:string">
                      <xs:attribute name="postalCode" use="optional">
                        <xs:simpleType>
                          <xs:restriction base="xs:unsignedShort">
                            <xs:pattern value="[0-9]{5}"/> <!--
Defining pattern for postal code with 5 digits-->
                          </xs:restriction>
                        </xs:simpleType>
                      </xs:attribute>
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>

```



```

    </xs:element>
    <xs:element name="phoneNumber" maxOccurs="unbounded">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:integer">
            <xs:attribute name="type" use="required">
              <xs:simpleType>
                <xs:restriction base="xs:string"> <!--
Defining restriction and enumeration for Phonenum type as mobile or telephone-
->
                  <xs:enumeration value="Telephone"/>
                  <xs:enumeration value="Mobile"/>
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>
    <xs:element name="website" type="xs:string" minOccurs="0" maxOccurs
="1"/>
    <xs:element name="logo"/>
    <xs:element name="email" type="xs:string" minOccurs="0" maxOccurs="
1"/>
    <xs:element name="managerName">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:string">
            <xs:attribute name="title" use="required">
              <xs:simpleType>
                <xs:restriction base="xs:string"> <!--
Defining restriction and enumeration for Manager title as Mr./Mrs./Ms.-->
                  <xs:enumeration value="Mr."/>
                  <xs:enumeration value="Mrs."/>
                  <xs:enumeration value="Ms."/>
                </xs:restriction>
              </xs:simpleType>
            </xs:attribute>
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
</xs:element>

```

```

<xs:element name="songs">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="song" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="genres">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="genre" maxOccurs="unbounded" type="xs:
:string" />
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="artists">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="artist" maxOccurs="unbounded">
                    <xs:complexType>
                      <xs:simpleContent>
                        <xs:extension base="xs:string">
                          <xs:attribute name="type" use="required">
                            <xs:simpleType>
                              <xs:restriction base="xs:string"> <!--
Defining restriction and enumeration for singer type-->
                                <xs:enumeration value="Singer"/>
                                <xs:enumeration value="Band"/>
                                <xs:enumeration value="Rapper"/>
                              </xs:restriction>
                            </xs:simpleType>
                          </xs:attribute>
                        </xs:extension>
                      </xs:simpleContent>
                    </xs:complexType>
                  </xs:element>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="director" type="xs:string" minOccurs="0"/>
            <xs:element name="producers" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="producer" type="xs:string" maxOccurs=
"unbounded"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

        </xs:complexType>
    </xs:element>
    <xs:element name="lyricists">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="lyricist" type="xs:string" maxOccurs=
"unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="album" minOccurs="0" type="xs:string"/>
    <xs:element name="releaseYear">
        <xs:simpleType>
            <xs:restriction base="xs:unsignedShort">
                <xs:pattern value="[0-9]{4}"/> <!--
Defining pattern for song release year as 2019-->
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <xs:element name="musicLabel" type="xs:string" minOccurs="0"
/>

    <xs:element name="length">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:pattern value="[0-9]{2}m:[0-9]{2}s"/> <!--
Defining pattern for song length as 05m:33s-->
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <xs:element name="price" type="xs:string"/>
</xs:sequence>
<xs:attribute name="songTitle" type="xs:string" use="required"/
>
    <xs:attribute name="language" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>

```

## CSS content

```
*{
  margin: 0;
  padding: 0;
  font-family: Century Gothic, sans-serif;
  letter-spacing: 1.5;
  color: white;
  display: block;
}

store{
  background-image: url(doodle.png);
  background-color: #5680E9;
  background-attachment: fixed;
  padding-left: 9%;
  padding-right: 9%;
  padding-top: 1%;
}

storeDetail{
  text-align: center;
  background: linear-gradient(#8860D0, transparent);
}

name{
  margin-top: auto;
  padding-top: 5%;
  font-family: Gotham;
  font-weight: lighter;
  font-size: 70px;
}

location{
  margin-top: -50%;
  padding-top: 10%;
  position: relative;
  display: inline-block;
  text-align: center;
  align-items: center;
  align-content: center;
}

location::before{
  content: icon;
```

```
}
location::after{
    content: ' (Postal Code: 'attr(postalCode)')';
    display: inline;
}

phoneNumber[type="Telephone"]::before{
    content: "Telephone Number: ";
}

phoneNumber[type="Mobile"]::before{
    content: "Mobile Number: ";
}

website{
    text-decoration: underline;
    color: rgb(0, 255, 42);
}

logo{
    position: relative;
    margin-top: -15%;
    margin-left: 46%;
    padding-top: -12%;
    padding-bottom: 10%;
    background-image: url(logo.png);
    height: 80px;
    width: 120px;
    background-repeat: no-repeat;
    display: block;
    cursor: pointer;
}

email::before{
    content: "E-mail: ";
}

managerName::before{
    content: attr(title) ' ';
}

managerName::after{
    content: " (Store Manager)";
}
```

```
managerName::after{
  margin-top: 60px;
  font-size: 40px;
  font-family: Roboto, sans-serif;
  display: block;
  content: "SONGS LIST";
  letter-spacing: 5pt;
}

songs{
  padding-left: 2%;
  display: list-item;
  width: 100%;
  height: 100%;
  display: inline-block;
  margin-top: -2%;
  padding-top: 50px;
  font-family: Verdana, Geneva, Tahoma, sans-serif;
}

song::before{
  content: attr(songTitle);
  font-weight: bold;
  font-size: xx-large;
  text-shadow: 2px 2px #8860D0;
}

song::after{
  padding-top: 1%;
  content: "Language: "attr(language);
  font-weight: lighter;
  font-size: medium;
  letter-spacing: 1pt;
  color: rgb(227, 227, 255);
}

song{
  box-sizing: border-box;
  width: 30%;
  height: 430px;
  border-radius: 30px;
  transition: 0.4s ease-in-out;
  display: inline-block;
  float: left;
  position: relative;
```

```
padding: auto 5px;
cursor: pointer;
background-color: #5AB9EA;
text-align: center;
padding-top: 2%;
margin-right: 50px;
margin-bottom: 50px;
}

song:hover{
  transform: translateY(10px);
  background-color: #8860D0;
}

genres::before{
  content: "Genre: ";
  letter-spacing: .5pt;
}

genre::after{
  content: ",";
}

genres{
  padding-top: 10px;
  display: list-item;
}

genre{
  display: inline-flex;
  font-weight: bold;
  font-size-adjust: ;
}

artists{
  display: list-item;
  padding: 5px 5px;
}

artist{
  display: inline-flex;
  font-weight: bold;
  font-size: large;
}
```

```
artist::after{
    content: ",";
}

artists::before{
    content: "Artist: ";
    letter-spacing: 1pt;
}

director::before{
    content: "Director: ";
    font-weight: normal;
    letter-spacing: 1pt;
}

director{
    font-weight: bold;
}

producers::before{
    content: "Producer: ";
    letter-spacing: 0.5pt;
}

producer{
    display: inline-flex;
    font-weight: bold;
}

producer::after{
    content: ",";
}

lyricists::before{
    content: "Lyricist: ";
    letter-spacing: 1pt;
}

lyricist{
    display: inline-flex;
    font-weight: bold;
}

lyricist::after{
    content: ",";
```



```
}

album::before{
    content: "Album: ";
    letter-spacing: 1pt;
    font-weight: normal;
}

album{
    font-weight: bold;
}

releaseYear::before{
    content: "Release Year: ";
    letter-spacing: 0.5pt;
    font-weight: normal;
}

releaseYear{
    font-weight: bold;
}

musicLabel::before{
    content: "Music Label: ";
    letter-spacing: 0.5pt;
    font-weight: normal;
}

musicLabel{
    font-weight: bold;
}

length::before{
    content: "Duration: ";
    letter-spacing: 0.5pt;
    font-weight: normal;
}

length{
    font-weight: bold;
}

price::before{
    content: "Price: ";
    letter-spacing: 0.5pt;
```

```

    font-size: x-large;
}

price{
    display: block;
    margin-left: 20%;
    margin-right: 20%;
    margin-top: 5%;
    margin-bottom: 5%;
    font-size: x-large;
    color: rgb(255, 255, 255);
    border: 2px solid white;
    border-radius: 30px;
}

```

## XML SCHEMA VS DTD

The difference between schema and DTD are:

XSD	DTD
Stands for XML Schema Definition.	Stands for Document Type Definition.
Written in XML.	Derived from SGML syntax.
As based on XML, data type and constraints for an element can be defined.	Weakly typed, constraints cannot be set.
44 data types are supported.	Only 10 data types are supported.
Support customized datatypes.	Does not support customized data type.
Supports namespaces completely but does not support entities.	Only supports namespace prefixes but supports entities.
Mixed content is easy to develop.	Mixed content is difficult to develop.

Defines order for child elements.	Does not define order for child elements.
Provides more control on XML structure.	Provides less control on XML structure.
Supports code reusability.	Does not support code reusability.
Simple to learn as there is no need to learn new language.	Harder than XSD and not simple to learn.
It is extensible.	It is not extensible.
More suitable for large XML data and data sets.	More suitable for small XML data.
Does not allow inline definition.	Allows inline definition.

*Table 1: Schema vs DTD*

## TESTING

### Test 1: WELL-FORMED

Objective	To test whether the written XML document is well-formed.
Action	XML document is properly written and structured with proper indentation and opened in browser without any stylesheet referenced.
Expected Result	All elements should be structured and nested properly having a closing tag, preserved whitespace, quoted attribute values, root element on top and tree-structured.
Actual Result	Well-formed
Conclusion	Test success.

*Table 2: Test case: well-formed XML document*

```

catalog_19033540.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <store>
3      <storeDetail>
4          <name>Divine Music Store</name>
5          <location postalCode="56600">Salakpur, Morang</location>
6          <phoneNumber type="Telephone">021545720</phoneNumber>
7          <phoneNumber type="Mobile">9819090917</phoneNumber>
8          <website>www.divinemusicstore.com.np</website>
9          <logo/>
10         <email>divinemusicstore@gmail.com</email>
11         <managerName title="Mr.">Susan Shrestha</managerName>
12     </storeDetail>
13
14     <songs>
15         <song songTitle="Sweet child O' mine" language="English">
16             <genres>
17                 <genre>Hard Rock</genre>
18                 <genre>Glam Metal</genre>
19             </genres>
20             <artists>
21                 <artist type="Band">Guns N' Roses</artist>
22             </artists>
23             <director>Nigel Dick</director>
24             <producers>
25                 <producer>Mike Clink</producer>
26             </producers>
27             <lyricists>
28                 <lyricist>Guns N' Roses</lyricist>
29             </lyricists>
30             <album>Appetite for Destruction</album>
31             <releaseYear>1988</releaseYear>
32             <musicLabel>Geffen</musicLabel>
33             <length>05m:02s</length>
34             <price>Rs.900</price>
35         </song>
36
37         <song songTitle="लेकाली चोयाको डोको" language="Nepali">
38             <genres>
39                 <genre>Folk</genre>
40             </genres>
41             <artists>
42                 <artist type="Singer">Ram Thapa</artist>
43             </artists>
44             <director>Sudan Sharma</director>
45             <producers>
46                 <producer>Music Nepal</producer>
47             </producers>
48             <lyricists>
49                 <lyricist>Sudan Sharma</lyricist>

```

Figure 2: Well-formed XML Code

```

▼<store>
  ▼<storeDetail>
    <name>Divine Music Store</name>
    <location postalCode="56600">Salakpur, Morang</location>
    <phoneNumber type="Telephone">021545720</phoneNumber>
    <phoneNumber type="Mobile">9819090917</phoneNumber>
    <website>www.divinemusicstore.com.np</website>
    <logo/>
    <email>divinemusicstore@gmail.com</email>
    <managerName title="Mr.">Susan Shrestha</managerName>
  </storeDetail>
  ▼<songs>
    ▼<song songTitle="Sweet child O' mine" language="English">
      ▼<genres>
        <genre>Hard Rock</genre>
        <genre>Glam Metal</genre>
      </genres>
      ▼<artists>
        <artist type="Band">Guns N' Roses</artist>
      </artists>
      <director>Nigel Dick</director>
      ▼<producers>
        <producer>Mike Clink</producer>
      </producers>
      ▼<lyricists>
        <lyricist>Guns N' Roses</lyricist>
      </lyricists>
      <album>Appetite for Destruction</album>
      <releaseYear>1988</releaseYear>
      <musicLabel>Geffen</musicLabel>
      <length>05m:02s</length>
      <price>Rs.900</price>
    </song>
    ▼<song songTitle="लेकाली चोयाको डोको" language="Nepali">
      ▼<genres>
        <genre>Folk</genre>
      </genres>
      ▼<artists>
        <artist type="Singer">Ram Thapa</artist>
      </artists>
      <director>Sudan Sharma</director>
      ▼<producers>
        <producer>Music Nepal</producer>
      </producers>
      ▼<lyricists>
        <lyricist>Sudan Sharma</lyricist>
      </lyricists>
      <album>Thokna Madal Thok</album>
      <releaseYear>2000</releaseYear>
      <musicLabel>Music Nepal</musicLabel>
      <length>02m:55s</length>
      <price>Rs.800</price>
    </song>
    ▼<song songTitle="I Want to Break Free" language="English">
      ▼<genres>
        <genre>Rock</genre>
        <genre>Synth-Pop</genre>
        <genre>Classic Rock</genre>
      </genres>
      ▼<artists>
        <artist type="Band">Queen</artist>
      </artists>
      ▼<producers>
        <producer>Queen</producer>
      </producers>
    </song>
  </songs>
</store>

```

Figure 3: Well-formed XML document loaded on browser

**Test 2: VALID (DTD)**

Objective	To test whether the XML document is valid when referenced to DTD file.
Action	<ul style="list-style-type: none"> <li>• catalog_19033540.dtd is referenced in XML document for validation.</li> <li>• An XML validation website (<a href="http://www.xmlvalidation.com">www.xmlvalidation.com</a>) is opened in browser and XML code is pasted and validate button is clicked.</li> <li>• Next the website asks to upload referenced DTD file, then catalog_19033540.dtd file is uploaded and continue validation is clicked.</li> </ul>
Expected Result	Validator should show no error for the XML document and DTD file.
Actual Result	No errors were found
Conclusion	Test success.

*Table 3: Test Case: Valid (DTD)***Please copy your XML document in here:**

```

<artist type=" Singer" >Chinmayi Chakraborty</artist>
</artists>
<director>Shankar Eshaan Loy</director>
<producers>
  <producer>T-Series</producer>
</producers>
<lyricists>
  <lyricist>Amitabh Bhattacharya</lyricist>
</lyricists>
<album>2 States</album>
<releaseYear>2014</releaseYear>
<musicLabel>T-Series</musicLabel>
<length>03m:43s</length>
<price>Rs.400</price>
</song>
</songs>
</store>

```

Or upload it:

 No file chosen

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 4: Uploading XML code for validation*

The file `catalog_19033540.dtd` is being referenced. Please copy it in here, so that the validation can continue:

```
<!ELEMENT store (storeDetail,songs)>

<!ELEMENT storeDetail (name,location,phoneNumber+,website?,logo,email?,managerName)>

<!ELEMENT songs (song)+>

<!ELEMENT name (#PCDATA)>

<!ELEMENT location (#PCDATA)>
<!-- ATTLIST location postalCode CDATA #IMPLIED -->

<!ELEMENT phoneNumber (#PCDATA)>
<!-- ATTLIST phoneNumber type (Telephone|Mobile) #REQUIRED -->

<!ELEMENT website (#PCDATA)>

<!ELEMENT logo EMPTY>
```

Or upload it:

No file chosen

Figure 5: Uploading DTD file for validation

## No errors were found

The following files have been uploaded so far:

[XML document:](#) 

[catalog\\_19033540.dtd](#) 

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

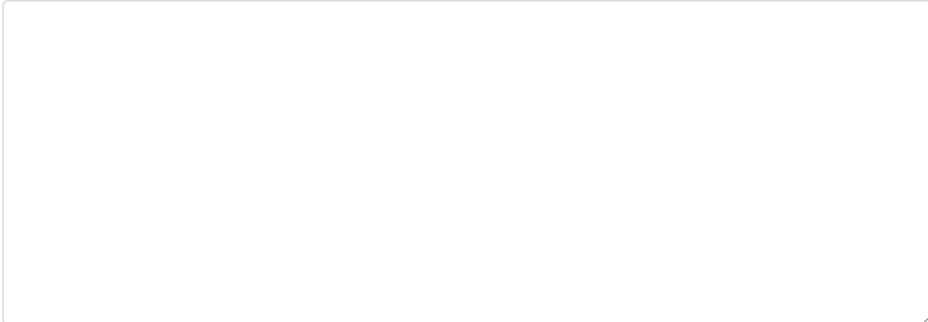
Figure 6: Successfully validated result

**Test 3: VALID (XSD)**

Objective	To test whether the XML document is valid when referenced to XSD file.
Action	<ul style="list-style-type: none"> <li>• catalog_19033540.xsd is referenced in XML document for validation.</li> <li>• An XML validation website (<a href="http://www.xmlvalidation.com">www.xmlvalidation.com</a>) is opened in browser and XML code is pasted and validate button is clicked.</li> <li>• Next the website asks to upload referenced XSD file, then catalog_19033540.xsd file is uploaded and continue validation is clicked.</li> </ul>
Expected Result	Validator should show no error for the XML document and XSD file.
Actual Result	No errors were found
Conclusion	Test success.

*Table 4: Test Case: Valid (XSD)*

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



Or upload it:

catalog\_19033540.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.

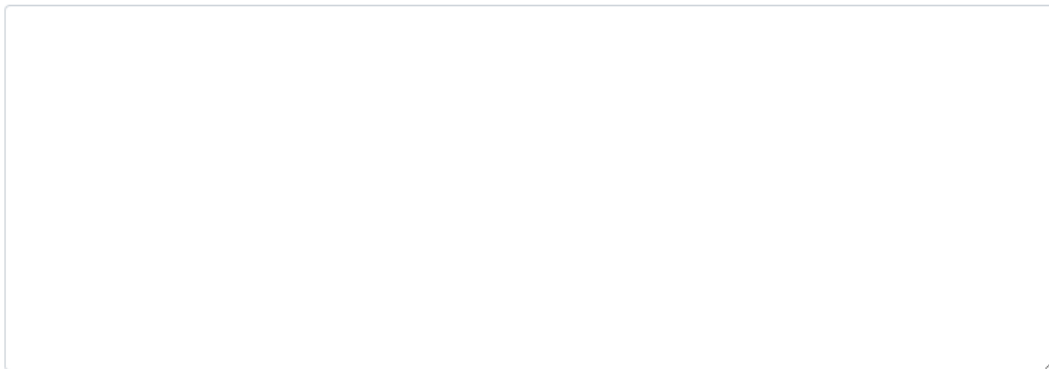
To validate the XML document against an external XML schema, click below.

☐ Validate against external XML schema

*Figure 7: Uploading XML file for validation*



The file catalog\_19033540.xsd is being referenced. Please copy it in here, so that the validation can continue:



Or upload it:

catalog\_19033540.xsd

The following files have been uploaded so far:

[XML document:](#) 

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

*Figure 8: Uploading XSD (Schema) file for validation*

## No errors were found

The following files have been uploaded so far:

[XML document:](#) 

[catalog\\_19033540.xsd](#) 

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

*Figure 9: Successfully validated XML Schema result*

**Test 4: VALID CSS LINKING**

Objective	To test and validate how the XML document is loaded in web browser after referencing CSS file.
Action	CSS referencing tag is written after <?xml?> tag and the XML document is opened in browser.
Expected Result	Browser should load the XML document structured with CSS.
Actual Result	Proper list of songs and store details are displayed with CSS designed view.
Conclusion	Test success.

*Table 5: Test Case: Valid CSS linking*

```

catalog_19033540.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3  <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd" -->
4  <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6  <!-- <store> -->
7      <storeDetail>
8          <name>Divine Music Store</name>
9          <location postalCode="56600">Salakpur, Morang</location>
10         <phoneNumber type="Telephone">021545720</phoneNumber>
11         <phoneNumber type="Mobile">9819090917</phoneNumber>
12         <website>www.divinemusicstore.com.np</website>
13         <logo/>
14         <email>divinemusicstore@gmail.com</email>
15         <managerName title="Mr.">Susan Shrestha</managerName>
16     </storeDetail>
17

```

*Figure 10: Linking CSS file to XML document*

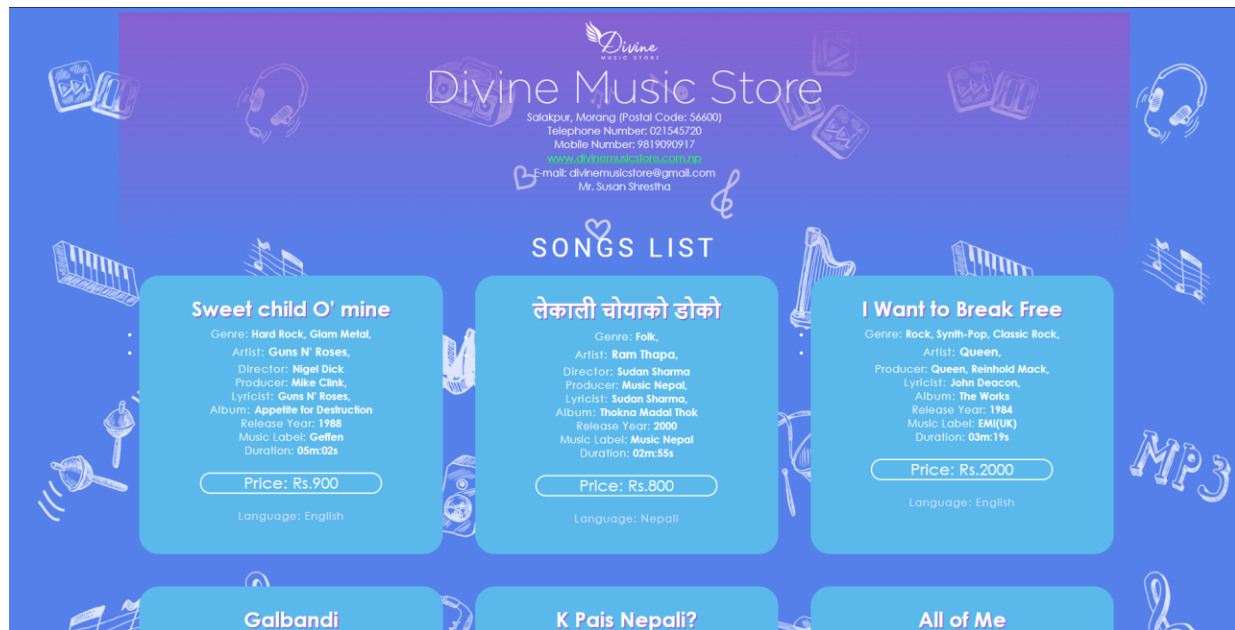


Figure 11: CSS linked XML document



Figure 12: CSS linked XML document

**Test 5: VALIDATING ENUMERATION**

Objective	To test whether the XML is valid and how the validator responds when values against enumeration is provided in XML document.
Action	In phoneNumber element, its type attribute value is changed to "Landline" instead of Telephone. And in managerName element, the title attribute value is entered as "Mister." Instead of "Mr."
Expected Result	Validator should display error result.
Actual Result	4 errors have been found.
Conclusion	Test success.

*Table 6: Test Case: Enumeration Validation*

```

catalog_19033540.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3  <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd" -->
4  <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6  <!-- <store> -->
7      <storeDetail>
8          <name>Divine Music Store</name>
9          <location postalCode="56600">Salakpur, Morang</location>
10         <phoneNumber type="Landline">021545720</phoneNumber>
11         <phoneNumber type="Mobile">9819090917</phoneNumber>
12         <website>www.divinemusicstore.com.np</website>
13         <logo/>
14         <email>divinemusicstore@gmail.com</email>
15         <managerName title="Mister.">Susan Shrestha</managerName>
16     </storeDetail>
17

```

*Figure 13: Changing enumerated attribute values*

## Please copy your XML document in here:

```
<storeDetail>
  <name>Divine Music Store</name>
  <location postalCode="56600">Salakpur, Morang</location>
  <phoneNumber type="Landline">021545720</phoneNumber>
  <phoneNumber type="Mobile">9819090917</phoneNumber>
  <website>www.divinemusicstore.com.np</website>
  <logo/>
  <email>divinemusicstore@gmail.com</email>
  <managerName title="Mister.">Susan Shrestha</managerName>
</storeDetail>
```

Or upload it:

catalog\_19033540.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 14: Uploading XML Code for enumeration validation

**The file catalog\_19033540.xsd is being referenced. Please copy it in here, so that the validation can continue:**

Or upload it:

catalog\_19033540.xsd



The following files have been uploaded so far:

[XML document:](#) 





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

Figure 15: Uploading Schema file for enumeration validation

**4 errors have been found!**



Click on  to jump to the error. In the document, you can point at  with your mouse to see the error message.

**Errors in the XML document:**

-  10:38 cvc-attribute.3: The value 'Landline' of attribute 'type' on element 'phoneNumber' is not valid with respect to its type, 'null'.
-  10:38 cvc-enumeration-valid: Value 'Landline' is not facet-valid with respect to enumeration '[Telephone, Mobile]'. It must be a value from the enumeration.
-  15:38 cvc-attribute.3: The value 'Mister.' of attribute 'title' on element 'managerName' is not valid with respect to its type, 'null'.
-  15:38 cvc-enumeration-valid: Value 'Mister.' is not facet-valid with respect to enumeration '[Mr., Mrs., Ms.]'. It must be a value from the enumeration.

**XML document:**

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3 <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd" -->
4 <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5 xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6 <!-- <store> -->
7   <storeDetail>
8     <name>Divine Music Store</name>
9     <location postalCode="56600">Salakpur, Morang</location>
10    <phoneNumber type="Landline">
11      021545720</phoneNumber>
12    <phoneNumber type="Mobile">9819090917</phoneNumber>
13    <website>www.divinemusicstore.com.np</website>
14    <logo/>
15    <email>divinemusicstore@gmail.com</email>
16    <managerName title="Mister.">
17      Susan Shrestha</managerName>
18  </storeDetail>

```

*Figure 16: Result of enumeration validation*

## Test 6: VALIDATING PATTERNED VALUES

Objective	To test whether the XML is valid and how the validator responds when values against set pattern is provided in XML document.
Action	In location element, its postalCode attribute value is changed to "1234567" instead of 5-digit postal code. And in releaseYear element, value is entered as "1988June." Instead of 4-digit year value.
Expected Result	Validator should display error result.
Actual Result	4 errors have been found.
Conclusion	Test success.

*Table 7: Test Case: Patterned values validation*

```

catalog_19033540.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3  <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd" -->
4  <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6  <!-- <store> -->
7  <storeDetail>
8      <name>Divine Music Store</name>
9      <location postalCode="1234567">Salakpur, Morang</location>
10     <phoneNumber type="Telephone">021545720</phoneNumber>
11     <phoneNumber type="Mobile">9819090917</phoneNumber>
12     <website>www.divinemusicstore.com.np</website>
13     <logo/>
14     <email>divinemusicstore@gmail.com</email>
15     <managerName title="Mr.">Susan Shrestha</managerName>
16 </storeDetail>
17
18 <songs>
19     <song songTitle="Sweet child O' mine" language="English">
20         <genres>
21             <genre>Hard Rock</genre>
22             <genre>Glam Metal</genre>
23         </genres>
24         <artists>
25             <artist type="Band">Guns N' Roses</artist>
26         </artists>
27         <director>Nigel Dick</director>
28         <producers>
29             <producer>Mike Clink</producer>
30         </producers>
31         <lyricists>
32             <lyricist>Guns N' Roses</lyricist>
33         </lyricists>
34         <album>Appetite for Destruction</album>
35         <releaseYear>1988June</releaseYear>
36         <musicLabel>Geffen</musicLabel>
37         <length>05m:02s</length>
38         <price>Rs.900</price>
39     </song>

```

Figure 17: Changing patterned value for validation

## Please copy your XML document in here:

```
xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
<!-- <store> -->
  <storeDetail>
    <name>Divine Music Store</name>
    <location postalCode="1234567">Salakpur, Morang</location>
    <phoneNumber type="Telephone">021545720</phoneNumber>
    <phoneNumber type="Mobile">9819090917</phoneNumber>
    <website>www.divinemusicstore.com.np</website>
    <logo/>
    <email>divinemusicstore@gmail.com</email>
```

Or upload it:

No file chosen

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 18: Uploading XML code for patterned value validation

**The file catalog\_19033540.xsd is being referenced. Please copy it in here, so that the validation can continue:**

Or upload it:

catalog\_19033540.xsd

The following files have been uploaded so far:




[XML document:](#) 





Figure 19: Uploading Schema file for patterned value validation



**4 errors have been found!**


Click on  to jump to the error. In the document, you can point at  with your mouse to see the error message.

**Errors in the XML document:**

-  9: 40 cvc-attribute.3: The value '1234567' of attribute 'postalCode' on element 'location' is not valid with respect to its type, 'null'.
-  9: 40 cvc-pattern-valid: Value '1234567' is not facet-valid with respect to pattern '[0-9]{5}' for type 'null'.
-  35: 48 cvc-pattern-valid: Value '1988June' is not facet-valid with respect to pattern '[0-9]{4}' for type 'null'.
-  35: 48 cvc-type.3.1.3: The value '1988June' of element 'releaseYear' is not valid.

**XML document:**

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3 <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd"> -->
4 <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5 xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6 <!-- <store> -->
7 <storeDetail>
8   <name>Divine Music Store</name>
   <location postalCode="1234567">
9      Salakpur, Morang</location>

```

*Figure 20: Result of patterned value validation*

## Test 7: REQUIRED ATTRIBUTE VALUE

Objective	To test whether the XML is valid and how validator responds when an element's attribute with use="required" is removed from XML file.
Action	In phoneNumber element, its attribute type is removed and XML is placed for validation.
Expected Result	Validator should show attribute error.
Actual Result	Attribute "type" must appear on element phoneNumber.
Conclusion	Test success.

*Table 8: Test Case: Required Attribute Value validation*

```

catalog_19033540.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3  <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd" -->
4  <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6  <!-- <store> -->
7      <storeDetail>
8          <name>Divine Music Store</name>
9          <location postalCode="56600">Salakpur, Morang</location>
10         <phoneNumber type="Telephone">021545720</phoneNumber>
11         <phoneNumber>9819090917</phoneNumber>
12         <website>www.divinemusicstore.com.np</website>
13         <logo/>
14         <email>divinemusicstore@gmail.com</email>
15         <managerName title="Mr.">Susan Shrestha</managerName>
16     </storeDetail>

```

Figure 21: Removing required type attribute

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

```

xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
<!-- <store> -->
<storeDetail>
  <name>Divine Music Store</name>
  <location postalCode="56600">Salakpur, Morang</location>
  <phoneNumber type="Telephone">021545720</phoneNumber>
  <phoneNumber>9819090917</phoneNumber>
  <website>www.divinemusicstore.com.np</website>
  <logo/>
  <email>divinemusicstore@gmail.com</email>

```

Or upload it:

No file chosen

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

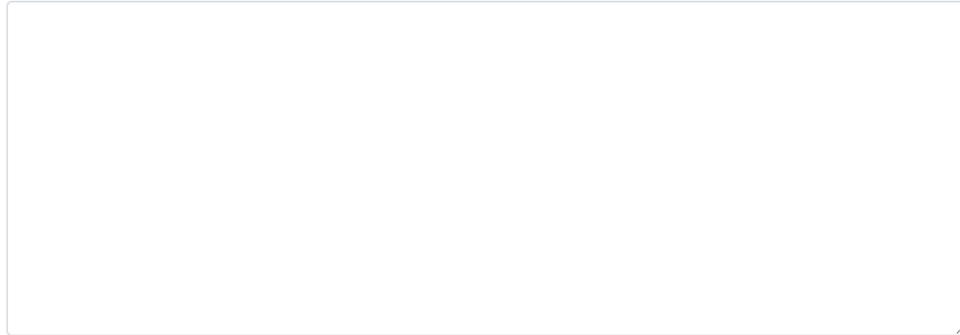
If neither an XML schema nor a DTD is declared, only a syntax check is performed.

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

☐ Validate against external XML schema

Figure 22: Uploading XML code


The file catalog\_19033540.xsd is being referenced. Please copy it in here, so that the validation can continue:



Or upload it:



catalog\_19033540.xsd

The following files have been uploaded so far:


[XML document](#) 

*Figure 23: Uploading Schema file for validating XML*

### An error has been found!


Click on  to jump to the error. In the document, you can point at  with your mouse to see the error message.

### Errors in the XML document:

 11:22 cvc-complex-type.4: Attribute 'type' must appear on element 'phoneNumber'.

### XML document:

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3 <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd"> -->
4 <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5 xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6 <!-- <store> -->
7   <storeDetail>
8     <name>Divine Music Store</name>
9     <location postalCode="56600">Salakpur, Morang</location>
10    <phoneNumber type="Telephone">021545720</phoneNumber>
11    <phoneNumber>
 9819090917</phoneNumber>

```

*Figure 24: Result of required attribute validation*

**Test 8: MATCHING TAG**

Objective	To test whether the XML is valid and how validator responds when an element's ending tag is removed or missing.
Action	In the XML document the matching end tag for <storeDetail> is removed and placed for validation.
Expected Result	Validator should show matching end tag error.
Actual Result	2 errors have been found including matching end-tag error and invalid content.
Conclusion	Test success.

*Table 9: Test Case: Matching Tag Validation*

```

catalog_19033540.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3  <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd" -->
4  <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6  <!-- <store> -->
7  <storeDetail>
8      <name>Divine Music Store</name>
9      <location postalCode="56600">Salakpur, Morang</location>
10     <phoneNumber type="Telephone">021545720</phoneNumber>
11     <phoneNumber type="Mobile">9819090917</phoneNumber>
12     <website>www.divinemusicstore.com.np</website>
13     <logo/>
14     <email>divinemusicstore@gmail.com</email>
15     <managerName title="Mr.">Susan Shrestha</managerName>
16
17
18     <songs>
19         <song songTitle="Sweet child O' mine" language="English">
20             <genres>
21                 <genre>Hard Rock</genre>

```

*Figure 25: Removing an ending tag from XML code*

## Please copy your XML document in here:

```
<storeDetail>
  <name>Divine Music Store</name>
  <location postalCode="56600">Salakpur, Morang</location>
  <phoneNumber type="Telephone">021545720</phoneNumber>
  <phoneNumber type="Mobile">9819090917</phoneNumber>
  <website>www.divinemusicstore.com.np</website>
  <logo/>
  <email>divinemusicstore@gmail.com</email>
  <managerName title="Mr.">Susan Shrestha</managerName>
```

Or upload it:

No file chosen

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 26: Uploading XML document for matching tag validation



**The file catalog\_19033540.xsd is being referenced. Please copy it in here, so that the validation can continue:**

Or upload it:

catalog\_19033540.xsd

Figure 27: Uploading Schema for matching tag validation

**2 errors have been found!**

Click on  to jump to the error. In the document, you can point at  with your mouse to see the error message.

**Errors in the XML document:**

- ✖ 18: 12 cvc-complex-type.2.4.d: Invalid content was found starting with element 'songs'. No child element is expected at this point.
- ✖ 347:3 The element type "storeDetail" must be terminated by the matching end-tag "</storeDetail>".

**XML document:**

```

1 <?xml version="1.0" encoding="utf-8"?>
2 <?xml-stylesheet type="text/css" href="catalog_19033540.css"?>
3 <!-- <!DOCTYPE store SYSTEM "catalog_19033540.dtd"> -->
4 <store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5 xsi:noNamespaceSchemaLocation="catalog_19033540.xsd">
6 <!-- <store> -->
7   <storeDetail>
8     <name>Divine Music Store</name>
9     <location postalCode="56600">Salakpur, Morang</location>
10    <phoneNumber type="Telephone">021545720</phoneNumber>
11    <phoneNumber type="Mobile">9819090917</phoneNumber>
12    <website>www.divinemusicstore.com.np</website>
13    <logo/>
14    <email>divinemusicstore@gmail.com</email>
15    <managerName title="Mr.">Susan Shrestha</managerName>
16
17
18    <songs>

```

*Figure 28: Result of matching tag validation*

## DEVELOPMENT PROCESS

The development process of this coursework begins with requirement analysis and observing possible specification required for the music store and naming the music store as “Divine Music Store”. After the detailed analysis from given scenario and observing specifications, all the possible elements for store details and songs are collected from which an XML tree diagram is developed using a diagram tool known as “draw.io” which made the development process much faster and efficient. In tree diagram, all the elements and their child elements with required and optional attributes and modifying symbols are drawn.

After completing the tree diagram, the next task involves writing an XML document named as catalog\_19033540.xml in which all elements and data are written by observing the tree structure. For writing the XML document, DTD and XSD “**Visual Studio Code**” IDE is used. As the xml file is written, it is checked if the XML document is well-formed by observing;

- All elements have closing tag properly nested.
- Spelling and cases are checked.
- Root element is checked.
- Whitespaces is preserved.
- Attributes values are quoted.

Next, a DTD file named catalog\_19033540.dtd is written for the XML file to define all elements, its modifying symbols, attributes, enumerated attribute type, PCDATA and CDATA.

Once the DTD is created, the next task is to write XML Schema Definition (XSD) named as catalog\_19033540.xsd for the XML document that defines all the elements, attributes, types and restrictions where required. The schema file is written using “Salami Slice” design also known as “Flat Catalog Design”. Salami Slice design contains all global elements that leads to many potential root elements. This design contains all reusable elements and supports reuse of elements from other documents. This design is implemented as the requirements of xml document including defining of

all elements and necessary attributes with restrictions or constraints are properly satisfied. Although this design has drawback that exposes the complexity in namespace and renders the root difficult to determine. Here in schema file patterns for elements and attributes are define using Regular Expresions (RegEx). For example, the postalCode element stores the postal code for the address with optional usage, for that the RegEx is defined as “[0-9]{5}”, this will only accept the integer value of 5-digits and other values are restricted or an error message is displayed. Likewise, enumerations are defined where required. For example, for the element phoneNumber it has an attribute named “type” which stores the type of phone number either “Mobile” or “Telephone”, which is exactly defined using enumeration in schema and other values except the defined will produce errors. Also, in the schema file data type for elements are defined using “type” attribute. Mostly used types in this coursework are xs:string, xs:integer and xs:unsignedShort. And for the occurrence of an element, “minOccurs” and “maxOccurs” attribute is used in element tag.

To display and render the XML data file in a web browser a CSS file is created and referenced in XML document. The CSS file styles the XML document that contains Store details in header section, a list of songs where each song has various details such as song title, artists, producer, lyricist, release year, price and so on. These details of song are displaying using floating box, border and the fonts such as “Century Gothic”, “Verdana”, “Geneva” and “Tahoma” and font sizes are style using different properties and font families such as “Sans Serif” to look the XML document attractive and well structured.

Finally completed with writing DTD, XSD and CSS file next task is to test the validness of XML document for which a website ([www.xmlvalidation.com](http://www.xmlvalidation.com)) is used where the XML document is uploaded and the linked schema or DTD file is uploaded too and the validation is done. The testing part included the testing of XML document if it is well formed, valid, CSS referenced XML document and other testing such as pattern values, mismatch end-tag, attribute values and enumeration value test. All the proofs of testing including screenshot are attached to the documentation and the proof of well written XML, DTD and XSD file are pasted to document as well.



## CRITICAL ANALYSIS AND CONCLUSION

Observing the whole work that I have completed is really satisfying to me as well as very informative for developing my knowledge and skills. The challenges and obstacles that I have surpassed through while doing this coursework was quite hard.

To begin with, the first task was to analyze the requirement specification for developing the work where the difficult task was to adding new elements and relating those with real-time. After that, producing an XML tree diagram with those collected elements and defining the occurrence, attributes for elements was the challenging task itself because the whole XML document is based on the prepared XML tree where the changes had to be mad frequently as the real-time data was different from expected. For example, firstly I drew the tree diagram having “genre” element with single occurrence, later while collecting the real-time songs list, I found to know that the song can have multiple genre such as the first song in the list “Sweet child O’ mine” is a song of “Hard Rock” and “Glam Metal” genre. Similarly, for some elements and attributes the same thing was repeated which made me stuck for a while.

After completing the tree diagram, it was much easier to write the XML document, as all elements and attributes and songs data was ready. By referring the tree diagram and songs data I wrote the XML document with more efficiency but I was facing the problem with well-formedness of the XML document. Later, I properly went through the well-formed XML guidelines and managed to indent the XML document properly where I fixed the whitespaces, proper quoting of attributes, spelling mis-match and proper end-tags. Likewise, when I started writing DTD file for created XML structure, it was easier to enlist the elements and attributes. So, the DTD was properly defined but for that first I read the college lecture slides clearly and watched some YouTube videos guide while writing it.

Next, to define the XML, the most difficult challenge I faced was choosing the Schema design style between Russian Doll, Salami Slice and Venetian Blind that made me stuck at this point of development. To solve this confusion, I had to go through a bit of research and had perception on XSD style. Firstly, I analyzed my XML document to

clearly understand the structure, data, elements, attributes and types, after that I chose the schema design style that fit my XML document and it was Salami Slice (Flat Catalog). Although the schema design style is quite lengthy compared to Venetian Blind but it is less lengthy than Russian doll as it had moderate nesting and less complex compared to Venetian Blind design. And the most important point for choosing this design style was it was suitable for the XML document as all the restrictions and type for elements and attributed could be easily defined in schema. After writing the Schema file, some errors were found in schema as well as XML document. To solve that I went through both XML and XSD again and found some type and constraints violation which were fixed immediately.

The next and final challenge that I faced during the development process was to write CSS for XML though it was not that difficult to write but it consumed much more time than I expected because the requirement for rendering and displaying the XML structure was little ambiguous for which I had to consult with my module lecturer and tutor. After clearing my doubts, I found some difficulties while writing the CSS such as not proper alignment of data, CSS attributes confusion and so on. After all, I finally completed the CSS and started the testing process. In testing, I first decided to plan the test cases and made table for recording the test done. During the testing of overall developed work, everything was done properly.

After completing the testing, I started documenting overall work and explained the development process. In documentation all proofs of work done are attached and presented. Lastly, I formatted the document structure properly as per the guidelines provided by the module tutor.

After all these challenges and obstacles faced, I got my work completed giving my best where I learned to not give up on obstacles hence enhancing my skills to solve problems related while developing XML document, DTD, XSD and CSS. I get to know about various way to define an XML schema, write DTD, attractive CSS and most importantly well-formed and valid XML document. To conclude, the most significant thing is that everything I have learned will be useful for my future and help me to get my skill to another level. I hope to use everything I learned from this work.

## REFERENCES

liam. (2016) *Extensible Markup Language (XML)* [Online]. Available at: <https://www.w3.org/XML/>.

techopedia. (n.d.) *What is DTD?* [Online]. Available from: <https://www.techopedia.com/definition/5228/document-type-definition-dtd>.

tutorialspoint. (n.d.) *XML - Overview* [Online]. Available from: [https://www.tutorialspoint.com/xml/xml\\_overview.htm](https://www.tutorialspoint.com/xml/xml_overview.htm).

w3schools. (n.d.) *XML Schema Tutorial* [Online]. Available from: [https://www.w3schools.com/xml/schema\\_intro.asp](https://www.w3schools.com/xml/schema_intro.asp).