



Module Code & Module Title

CS5004NT Emerging Programming Platforms and Technologies

Assessment Weightage & Type

30% Individual Coursework

Year and Semester

2021 Spring

Student Name: Samuel Sherpa

London Met ID: 19031860

College ID: np05cp4a190148

Assignment Due Date: 7 May 2021

Assignment Submission Date: Week 20

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.

Acknowledgement

Life is not fair for everyone, but in every situation, I would like to open my heart and life in a fresh way where I surrender to acknowledge God the almighty in all my ways that his hand is upon me because of which I am able to study in this prestigious college and acquire to get more knowledge and skills.

I would like to express my special thanks of gratitude to my module lecturer MR. Pradhumna Dhungana for his guidance and support in completing my coursework. His calm nature and interesting subject-related conversations led me to ignite the curiosity of doing cool stuff with XML which helped me a lot to know more about the module and coursework.

I would like to acknowledge my parents and sister for continuous support in a home by providing me a peaceful environment where I could study very well. Last but not least I would like to thank all my classmates who have helped me a lot.

Thank you all from the bottom of my heart. God bless you all.

Abstract

This is good documentation of 30% individual coursework(CW) assessment of 'Emerging Programming Platforms and Technologies' submitted by Samuel Sherpa (UID: 19031860).

In this individual coursework, we students were treated as an XML developer who have to model a system for a hypothetical music store, 'Jetkhako Music Store (JMS)' after an interview with the store manager. The CW was intended to revolve around XML, DTD, and Schema but many other important topics were discussed below in the documentation. Let me write it down in points for easiness to read:-

- First of all XML file named catalog_19031860.xml was created for JMS which included information about its name, address, contact info, songs available in it with much detail such as singer name, awards, composers, and many more
- Document Type Defination (DTD) and Schema files named catalog_19031860.dtd and catalog_19031860.xsd was created to structure and validate above created xml file.
- CSS file was created and combined with an XML document to have a professional view of the XML file in a browser.
- The documentation part of CW sums up all information about students' critical views, knowledge depth of DTD, Schema, XML, and testing professionally created software.

Table of content

Contents

1.Introduction	1
Xml	1
DTD	1
XSD	1
Aims & Objectives.....	2
Aims	2
Objective	2
Data List/Tree Diagram	3
XML.....	6
Well defined CSS	16
DTD.....	18
Schema	20
Report	26
Testing.....	26
TEST 1: XML validation without using CSS	26
TEST 2: XML validation with DTD.....	28
TEST 3: XML validation with Schema	30
TEST 4: XML validation with CSS.....	32
TEST 5: XML validation with DTD for Modifying symbols.	34
TEST 6: XML validation with Schema for indicators.....	36
TEST 7: XML validation with Schema for data type	38
TEST 8: XML validation with Schema for Restriction of Gender	40
TEST 9: XML validation with Schema for Restriction of Telephone number	42

TEST 10: XML validation with Schema for Unique ID attribute.	44
Difference between Schema and DTD	46
How the coursework was developed?	49
Critical Analysis	54
Conclusion	56
Appendix (Optional):.....	Error! Bookmark not defined.
Venetian Blind:	Error! Bookmark not defined.
History of XML	Error! Bookmark not defined.
References.....	57

Table of figure

Figure 1 Tree diagram.....	3
Figure 2 Test 1	27
Figure 3 Test2. Uploading XML file in xmlvalidation.com	28
Figure 4 Test2. Uploading DTD file to xmlvalidation.com as required.	29
Figure 5 Test2. No errors found in above XML validation with DTD.....	29
Figure 6 Test3. Uploading XML file in xmlvalidation.com	30
Figure 7 Test3. Uploading XSD file in xmlvalidation.com as requested.	31
Figure 8 Test3. No errors found in XML validation with XSD	31
Figure 9 Test4. XML file without CSS.....	32
Figure 10 Test4. XML file with CSS.....	33
Figure 11 Test5 XML validation of Modifying Symbol test with DTD	34
Figure 12 Test5. Uploading XML document in xmlvalidation.com	35
Figure 13 Test5. Uploading DTD Document for validation.	35
Figure 14 Test6. XML validation for testing indicator using Schema	36
Figure 15 Test6. Error shown by Schema for breaking the rule of indicator minoccurs.	37
Figure 16 Test7. Validating XML for data type using Schema.....	38
Figure 17 Test7. Error display for data type in XML using Schema	39
Figure 18 Test8. Validating XML using Schema for restriction.....	40
Figure 19 Test8. Error displayed for restriction.	41
Figure 20 Test9. Validating Telephone element of XML.....	42
Figure 21 Test9. Error by Schema for Telephone	43
Figure 22 Test10. XML validation for Unique ID using Schema	44
Figure 23 Test 10. Error displayed due to same ID	45
Figure 24 Draw.io tool	50
Figure 25 Visual Studio code tool.....	50
Figure 26 xmlvalidation.com tool.....	51
Figure 27 chrome tool	51
Figure 28 KightShot tool.....	52
Figure 29 MSWord tool.	52
Figure 30 Easycodeformatter	53

Table of tables

Table 1 Elements of Tree diagram.	4
Table 2 Test 1 tesing table	26
Table 3 Test2 testing table.	28
Table 4 Test 3 testing table.	30
Table 5 Test4. XML with and without CSS.	32
Table 6 Test 5 testing XML modification using Schema.....	34
Table 7 Test6 validating indicator of xml using Schema.....	36
Table 8 Test7 validating data type of XML using Schema.....	38
Table 9 Test8. Testing restriction if XML using Schema	40
Table 10 Test9. Testing telephone restriction of XML	42
Table 11 Test 10 validating unique Id of XML Using Schema	44
Table 12 Difference between DTD and Schema	48

1.Introduction

Xml

Before I head to explain anything about ML, first to know about XML is, it's not a programming language but a formatting language used to format data or content usually for storage, transmission, and processing by a program. It tells us nothing about what we should do with the data like other programming languages, but in fact, many programming languages use, XML to structure and transfer data.

It's a markup language that has rules and tags that define the data and format of text while presenting text. XML is reusable because of its extensibility. The extensible word means it is meta language. Therefore, it can extend its specifications such as ADML, GML, WML, Math ML to define new markups.

DTD

It stands for Document Type Definition. It describes the tree structure of a document because it has a set of markup affirmations that actually define a type of document for the SGML -family markup language (GML, SGML, XML, HTML). Talking about DTD for xml then it defines structure, legal elements and attributes of an XML document. We have privilege of defining DTD either inside the (XML)instance document or in a separate document. (geeksforgeeksDTD, 2021)

XSD

It stands for XML Schema Definition. It simply defines every building block of an XML document which are the elements, attributes, child elements, data type of elements and attributes, and also default or fixed values of elements and attributes. XSD elements support Namespaces. XSD is a powerful XML editor which not only edits but also secures data communication. (Allaboutxml, 2021)

Aims & Objectives

Aims, objectives and goals are often considered same, scholars in education take the words aim and objective more narrowly and consider them very differently. We take aim as an end result but objective questions how we could achieve those results. Aims and objectives of this coursework is as follows:

Aims

- Learn the very basics of XML, DTD, and Schema.
- Learn the above topics in such a way, that we could explain them to a 6th-grade kid.
- After learning, model a system for a music store as an XML developer
- Create, XML, DTD, Schema, and CSS files for it.
- Make the best documentation ever of the above-done work.

Objective

- Revise all the lecture slides and start creating dummy files since we know 'Early birds get the worm first'.
- Ask confusion, error, or any problem to the module teacher.
- Reach out to friends for help, if so needed.
- Use the best tool or IDE to create XML files.
- Ask useful tips and tricks from seniors.

Data List/Tree Diagram

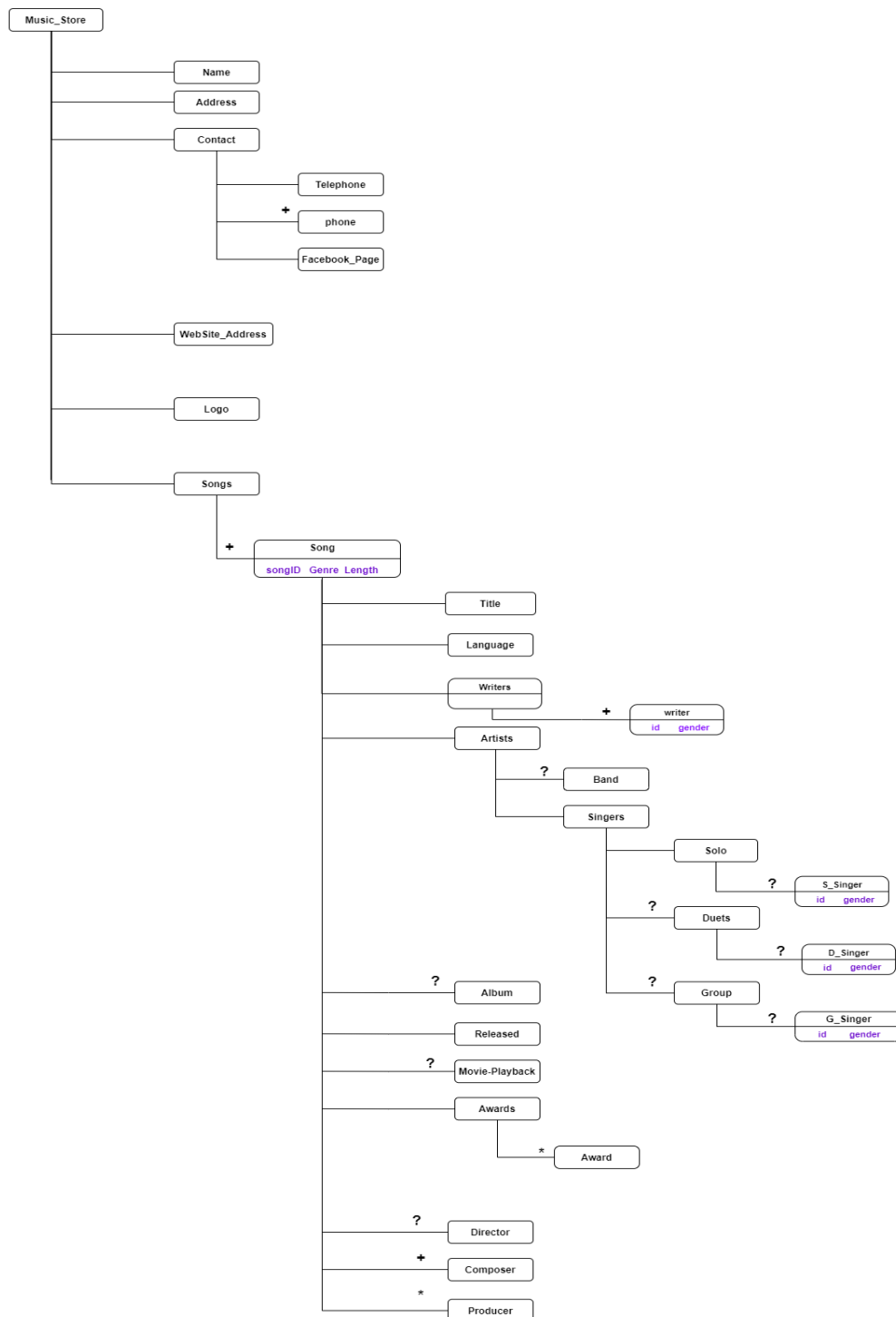


Figure 1 Tree diagram

Above figure represents tree diagram for building xml of given music store. Above tree diagram contains 21 data, 11 attributes and 13 optional elements. The data, attributes and optional data are as follows:

S.N.	Data	Attributes	Optional elements
1	Name	song_id	Phone
2	Address	Genre	Band
3	Telephone (+)	Length	Solo
4	Phone (+)	wr_id	S-Singer
5	Facebook_Page	wr_gender	Duets
6	Website_address	ssing_id	D-Singer
7	Logo	s_gender	Group
8	Title	dsing_id	G_Singer
9	Language	d_gener_id	Album
10	Writer (+)	gsing_id	Movie_Playback
11	Band (?)	g_gender	Award
12	S_Singer (?)		Director
13	D_Singer (?)		Producer
14	G_Singer (?)		
15	Album		
16	Released		
17	Movie_Playback		
18	Award (+)		
19	Director (?)		
20	Composer (+)		
21	Producer (*)		

Table 1 Elements of Tree diagram.

The signs present in the above figure are:

- **?** :It means that element can either be absent or appear for one time.
- *****: It means that element can either be absent or be present for multiple times.
- **+**: It means that element can appear for more than one time.

XML

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako
   Music Store"?>
3. <!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->
4.
5. <Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">
6.
7.     <Name>Jethako Music Store</Name>
8.     <Address>Damak-10, Jhapa, Nepal</Address>
9.     <Contact>
10.         <Telephone>023590972</Telephone>
11.         <Phone>9814978419</Phone>
12.         <Phone>9816924453</Phone>
13.         <Facebook_Page>Jethako Pasal</Facebook_Page>
14.     </Contact>
15.     <Website_address>www.Jethakopasal.com</Website_address>
16.     <Logo></Logo>
17.
18.     <!-- Song 1 -->
19.     <Songs>
20.         <Song song_id="song_1" Genre="AlternativeIndie" Length="4:22">
21.             <Title>Tum Hi Ho</Title>
22.             <Language>Hindi</Language>
23.             <Writers>
24.                 <writer wr_id="wr_1" wr_gender="Male">Mithun Sharma</writer>
25.             </Writers>
26.             <Artists>
27.                 <Band></Band>
28.                 <Singers>
29.                     <Solo>
30.                         <S_Singer ssing_id="sing_1" s_gender="Male">Arijit
Singh</S_Singer>
31.                     </Solo>

```

```

32.         </Singers>
33.     </Artists>
34.     <Album>Aashiqui 2</Album>
35.     <Released>16 March 2013</Released>
36.     <Movie_Playback>Aashiqui 2</Movie_Playback>
37.     <Awards>
38.         <Award>Filmfare Award for Best Male Playback Singer</Award>
39.         <Award>Best Music Director</Award>
40.         <Award>Best Lyrics 2014</Award>
41.         <Award>Most Entertaining Song</Award>
42.         <Award>Song of the year(2014)</Award>
43.         <Award>6th Mirchi Music Awards</Award>
44.     </Awards>
45.     <Director>Mithun Sharma</Director>
46.     <Composer>Mithun Sharma</Composer>
47.     <Producer>Mukesh Bhatt and Team</Producer>
48. </Song>
49.
50. <!-- Song 2 -->
51.
52. <Song song_id="song_2" Genre="Rock" Length="5:45">
53.     <Title>Kimi no nawa</Title>
54.     <Language>Japanese</Language>
55.     <Writers>
56.         <writer wr_id="wr_2" wr_gender="Male">Uzumdamu Seiju</writer>
57.     </Writers>
58.     <Artists>
59.         <Band>Radwips</Band>
60.         <Singers>
61.             <Solo>
62.                 <S_Singer ssing_id="sing_2" s_gender="Male">Yojiro
Noda</S_Singer>
63.             </Solo>
64.         </Singers>
65.     </Artists>
66.     <Album>Your Name</Album>
67.     <Released>24 August 2016</Released>

```

```

68.         <Movie_Playback>Your name</Movie_Playback>
69.         <Awards>
70.             <Award>58th Japan record Awards</Award>
71.             <Award>Japan Academy prize</Award>
72.             <Award>Japan Gold Disc Award</Award>
73.         </Awards>
74.         <Director>Radwimps</Director>
75.         <Composer>Radwimps</Composer>
76.         <Producer></Producer>
77.     </Song>
78.
79.
80.     <!-- Song 3 -->
81.
82.     <Song song_id="song_3" Genre="Nepali Folk" Length="6:28">
83.         <Title>Kusume Rumal</Title>
84.         <Language>Nepali</Language>
85.         <Writers>
86.             <writer wr_id="wr_3" wr_gender="Male">Tulsi Ghimire</writer>
87.         </Writers>
88.         <Artists>
89.             <Band></Band>
90.             <Singers>
91.                 <Duets>
92.                     <D_Singer dsing_id="sing_3" d_gender="Male">Udit
Narayan Jhap</D_Singer>
93.                     <D_Singer dsing_id="sing_4" d_gender="Female">Deepa
Jha</D_Singer>
94.                 </Duets>
95.             </Singers>
96.         </Artists>
97.         <Album></Album>
98.         <Released>1985</Released>
99.         <Movie_Playback>Kusume Rumal</Movie_Playback>
100.        <Awards>
101.            <Award>Most Entertaining Song</Award>
102.            <Award>Precious Song of the century</Award>

```

```

103.          </Awards>
104.          <Director>Ranjit Gazmer</Director>
105.          <Composer>Abhisek pradhan</Composer>
106.          <Producer></Producer>
107.      </Song>
108.
109.
110.      <!-- Song 4 -->
111.
112.      <Song song_id="song_4" Genre="Pop,rock" Length="3:36">
113.          <Title>Summer of 69</Title>
114.          <Language>English</Language>
115.          <Writers>
116.              <writer wr_id="wr_4" wr_gender="Male">Jim Vallance</writer>
117.              <writer wr_id="wr_04" wr_gender="Male">Bryan Adams</writer>
118.          </Writers>
119.          <Artists>
120.              <Band></Band>
121.              <Singers>
122.                  <Solo>
123.                      <S_Singer ssing_id="sings_4" s_gender="Male">Bryan
Adams</S_Singer>
124.                  </Solo>
125.              </Singers>
126.          </Artists>
127.          <Album>Reckless</Album>
128.          <Released>17 June 1985</Released>
129.          <Movie_Playback></Movie_Playback>
130.          <Awards>
131.              <Award>BMI (Broadcast Music Inc.)</Award>
132.              <Award>Procan Award</Award>
133.              <Award>Socan Classic Award</Award>
134.          </Awards>
135.          <Director></Director>
136.          <Composer>Bob clearmountain</Composer>
137.          <Composer>Bryan Adams</Composer>
138.          <Producer></Producer>

```



```
139.         </Song>
140.
141.
142.         <!-- Song 5 -->
143.
144.         <Song song_id="song_5" Genre="Nepali indie" Length="4:47">
145.             <Title>Sombadhan Timilai</Title>
146.             <Language>Nepali</Language>
147.             <Writers>
148.                 <writer wr_id="wr_5" wr_gender="Male">Biswas Lama</writer>
149.             </Writers>
150.             <Artists>
151.                 <Band>1974 AD</Band>
152.                 <Singers>
153.                     <Solo>
154.                         <S_Singer ssing_id="sing_5" s_gender="Male">Adrain
Pradhan</S_Singer>
155.                     </Solo>
156.                 </Singers>
157.             </Artists>
158.             <Album>Satabdi</Album>
159.             <Released>2001</Released>
160.             <Movie_Playback></Movie_Playback>
161.             <Awards>
162.                 <Award>Most Air Played</Award>
163.             </Awards>
164.             <Director>Manoj Kumar KC</Director>
165.             <Composer>Biswas Lama</Composer>
166.             <Composer>Adrain Pradhan</Composer>
167.             <Producer></Producer>
168.         </Song>
169.
170.
171.         <!-- Song 6 -->
172.
173.         <Song song_id="song_6" Genre="Hip hop,Pop-rap" Length="3:49">
174.             <Title>See you again</Title>
```

```

175.          <Language>English</Language>
176.          <Writers>
177.              <writer wr_id="wr_61" wr_gender="Male">Cameron</writer>
178.              <writer wr_id="wr_62" wr_gender="Male">Charlie Puth</writer>
179.              <writer wr_id="wr_63" wr_gender="Male">Thomaz Andrew
                Cedar</writer>
180.              <writer wr_id="wr_64" wr_gender="Male">Dann Hume</writer>
181.              <writer wr_id="wr_65" wr_gender="Male">Josh Hardy</writer>
182.          </Writers>
183.          <Artists>
184.              <Band></Band>
185.              <Singers>
186.                  <Duets>
187.                      <D_Singer dsing_id="sing_6" d_gender="Male">Charlie
                        Puth</D_Singer>
188.                      <D_Singer dsing_id="sing_61" d_gender="Male">Wiz
                        Khalifa</D_Singer>
189.                  </Duets>
190.              </Singers>
191.          </Artists>
192.          <Album></Album>
193.          <Released>10 March 2015</Released>
194.          <Movie_Playback>Fast & Furious</Movie_Playback>
195.          <Awards>
196.              <Award>BillBoard Hot 100</Award>
197.              <Award>Song of the year 2016</Award>
198.              <Award>Collaboration of the year</Award>
199.          </Awards>
200.          <Director></Director>
201.          <Composer>DJ Frank E</Composer>
202.          <Composer>Charlie Puth</Composer>
203.          <Composer>Andrew Cedar</Composer>
204.          <Producer></Producer>
205.      </Song>
206.
207.
208.      <!-- Song 7 -->

```

```

209.
210.      <Song song_id="song_7" Genre="Nepali Patriotic Folk" Length="1:11">
211.          <Title>Sayaun Thunga Phulka</Title>
212.          <Language>Nepali</Language>
213.          <Writers>
214.              <writer wr_id="wr_7" wr_gender="Male">Byakul Maila</writer>
215.          </Writers>
216.          <Artists>
217.              <Band></Band>
218.              <Singers>
219.                  <Solo>
220.                      <S_Singer ssing_id="sing_7" s_gender="Male">Amber
Gurung</S_Singer>
221.                  </Solo>
222.              </Singers>
223.          </Artists>
224.          <Album></Album>
225.          <Released>3 August 2007</Released>
226.          <Movie_Playback></Movie_Playback>
227.          <Awards>
228.              <Award>Third listed in Rio 2016</Award>
229.          </Awards>
230.          <Director></Director>
231.          <Composer>Amber Gurung</Composer>
232.          <Producer></Producer>
233.      </Song>
234.
235.
236.      <!-- Song 8 -->
237.
238.      <Song song_id="song_8" Genre="Feature Film soundtrack"
Length="6:41">
239.          <Title>Dil Hain Tumhara</Title>
240.          <Language>Hindi</Language>
241.          <Writers>
242.              <writer wr_id="wr_8" wr_gender="Male">Aseem Sinha</writer>
243.          </Writers>

```

```

244.          <Artists>
245.              <Band></Band>
246.              <Singers>
247.                  <Group>
248.                      <G_Singer gsing_id="sing_81" g_gender="Male">Udit
                          Narayan Jha</G_Singer>
249.                      <G_Singer gsing_id="sing_82" g_gender="Female">Alka
                          Yagni</G_Singer>
250.                      <G_Singer gsing_id="sing_83" g_gender="Male">Kumar
                          Sanu</G_Singer>
251.                  </Group>
252.              </Singers>
253.          </Artists>
254.          <Album></Album>
255.          <Released>28 June 2002</Released>
256.          <Movie_Playback>Dil Hain Tumhara</Movie_Playback>
257.          <Awards>
258.              <Award>Song of the year</Award>
259.              <Award>Mirchi Award 2994</Award>
260.          </Awards>
261.          <Director></Director>
262.          <Composer>Surinder Sodhi</Composer>
263.          <Producer>Nadeem Shravan</Producer>
264.      </Song>
265.
266.
267.      <!-- Song 9 -->
268.
269.      <Song song_id="song_9" Genre="Show tune" Length="3:45">
270.          <Title>Let it go</Title>
271.          <Language>English</Language>
272.          <Writers>
273.              <writer wr_id="wr_9" wr_gender="Female">Kristen Anderson-
                          Lopez</writer>
274.              <writer wr_id="wr_91" wr_gender="Male">Robert Lopez</writer>
275.          </Writers>
276.          <Artists>

```

```

277.          <Band></Band>
278.          <Singers>
279.              <Solo>
280.                  <S_Singer ssing_id="sing_9" s_gender="Female">Idina
Menzel</S_Singer>
281.              </Solo>
282.          </Singers>
283.      </Artists>
284.      <Album></Album>
285.      <Released>2013</Released>
286.      <Movie_Playback>Let it go</Movie_Playback>
287.      <Awards>
288.          <Award>walt Disney song of the year</Award>
289.      </Awards>
290.      <Director>Samuel aprehs</Director>
291.      <Composer>Kristen Anderson-Lopez</Composer>
292.      <Composer>Christophe Beck</Composer>
293.      <Producer></Producer>
294.  </Song>
295.
296.
297.      <!-- Song 10 -->
298.
299.      <Song song_id="song_10" Genre="Nepali folk Rock" Length="3:22">
300.          <Title>Tal ko pani</Title>
301.          <Language>Nepali</Language>
302.          <Writers>
303.              <writer wr_id="wr_10" wr_gender="Male">Anonymous</writer>
304.          </Writers>
305.          <Artists>
306.              <Band>Nepathya</Band>
307.              <Singers>
308.                  <Solo>
309.                      <S_Singer ssing_id="sing_10" s_gender="Male">Amrit
Gurung</S_Singer>
310.                  </Solo>
311.              </Singers>

```

```
312.          </Artists>
313.          <Album>Bheda ko Ooon Jasto</Album>
314.          <Released>2003</Released>
315.          <Movie_Playback></Movie_Playback>
316.          <Awards>
317.              <Award>Nepali heart song</Award>
318.          </Awards>
319.          <Director></Director>
320.          <Composer>Nepathya</Composer>
321.          <Producer>Nepathya</Producer>
322.      </Song>
323.  </Songs>
324. </Music_Store>
325.
```

Well defined CSS

```
1. Music_Store {
2.     font-family:sans-serif;
3.     font-size: 20px;
4.     margin-top: 10px;
5.     text-align: center;
6.     color: rgb(196, 44, 102);
7.     background-color: #41403f;
8. }
9.
10. Music_Store Name {
11.     font-family: initial;
12.     font-weight: bold;
13.     font-size: 30px;
14. }
15.
16. Logo {
17.     background-image: url("MusicShop_logo.png");
18.     width: 200px;
19.     height: 200px;
20.     background-repeat: no-repeat;
21.     position: relative;
22.     top: 10px;
23.     left: 10px;
24.     float: center;
25.     margin: -125px 0px 0px 0px;
26.     background-size: 200px 200px;
27.     border-radius: 10px;
28.     clip-path: circle();
29. }
30.
31.
32.
33. Song{
34.     display: list-item;
```

```
35.    list-style-type:decimal;
36.    font-family: Verdana, Geneva, Tahoma, sans-serif;
37.    font-size: 16px;
38.    color: rgb(255, 255, 255);
39.    margin: 20px 400px;
40.    padding: 10px 30px;
41.    background-color: #5ba6d1;
42.    border: solid 3px rgb(224, 22, 99);
43.    border-radius:8%;
44.    border-style:dotted;
45.
46. }
47.
48. Title {
49.    display:block;
50.    font-weight: bold;
51.    color: rgb(245, 184, 71);
52. }
53. Name,Address, Contact,Facebook_Page, Website_address,Logo{
54.    display: block;
55. }
56.
57. Language,Writers,Artists,Album,Released,Awards,Director,Composer,Producer {
58.    display: block;
59. }
60. Artists Band,Singers,Solo,Singer,Duets,Group{
61.    display: block;
62.    color: rgb(90, 250, 90);
63. }
64. Awards Award{
65.    display: block;
66.    color: yellow;
67. }
68.
```


DTD

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!ELEMENT Music_Store (Name,Address,Contact,Website_address,Logo,Songs)>
3. <!ELEMENT Name (#PCDATA)>
4. <!ELEMENT Address (#PCDATA)>
5. <!ELEMENT Contact (Telephone,Phone+,Facebook_Page)>
6. <!ELEMENT Telephone (#PCDATA)>
7. <!ELEMENT Phone (#PCDATA)>
8. <!ELEMENT Facebook_Page (#PCDATA)>
9. <!ELEMENT Website_address (#PCDATA)>
10.<!ELEMENT Logo (#PCDATA)>
11.<!ELEMENT Songs (Song+)>
12.<!ELEMENT Song
    (Title,Language,Writers,Artists,Album,Released,Movie_Playback,Awards,Director*
    ,Composer+,Producer*)>
13.<!ATTLIST Song song_id ID #REQUIRED>
14.<!ATTLIST Song Genre CDATA #REQUIRED>
15.<!ATTLIST Song Length CDATA #REQUIRED>
16.<!ELEMENT Title (#PCDATA)>
17.<!ELEMENT Language (#PCDATA)>
18.<!ELEMENT Writers (writer+)>
19.<!ELEMENT writer (#PCDATA)>
20.<!ATTLIST writer wr_id ID #REQUIRED>
21.<!ATTLIST writer wr_gender (Male | Female) #REQUIRED>
22.<!ELEMENT Artists (Band?,Singers)>
23.<!ELEMENT Band (#PCDATA)>
24.<!ELEMENT Singers (Solo?,Duets?,Group?)>
25.<!ELEMENT Solo (S_Singer?)>
26.<!ELEMENT S_Singer (#PCDATA)>
27.<!ATTLIST S_Singer ssing_id ID #REQUIRED>
28.<!ATTLIST S_Singer s_gender (Male | Female) #REQUIRED>
29.<!ELEMENT Duets (D_Singer*)>
30.<!ELEMENT D_Singer (#PCDATA)>
31.<!ATTLIST D_Singer dsing_id ID #REQUIRED>
32.<!ATTLIST D_Singer d_gender (Male | Female) #REQUIRED>

```

```
33.<!ELEMENT Group (G_Singer*)>
34.<!ELEMENT G_Singer (#PCDATA)>
35.<!ATTLIST G_Singer gsing_id ID #REQUIRED>
36.<!ATTLIST G_Singer g_gender (Male | Female) #REQUIRED>
37.<!ELEMENT Album (#PCDATA)>
38.<!ELEMENT Released (#PCDATA)>
39.<!ELEMENT Movie_Playback (#PCDATA)>
40.<!ELEMENT Awards (Award*)>
41.<!ELEMENT Award (#PCDATA)>
42.<!ELEMENT Director (#PCDATA)>
43.<!ELEMENT Composer (#PCDATA)>
44.<!ELEMENT Producer (#PCDATA)>
45.
```

Schema

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
3.     <xs:element name="Music_Store">
4.         <xs:complexType>
5.             <xs:sequence>
6.                 <xs:group ref="Music_StoreGR" />
7.             </xs:sequence>
8.         </xs:complexType>
9.     </xs:element>
10.
11.     <xs:complexType name="ContactType">
12.         <xs:sequence>
13.             <xs:group ref="ContactTypeGR" />
14.         </xs:sequence>
15.     </xs:complexType>
16.
17.     <xs:complexType name="SongsType">
18.         <xs:sequence>
19.             <xs:group ref="SongsTypeGR" />
20.         </xs:sequence>
21.     </xs:complexType>
22.
23.     <xs:complexType name="SongType">
24.         <xs:sequence>
25.             <xs:group ref="SongTypeGR" />
26.         </xs:sequence>
27.         <xs:attributeGroup ref="SongTypeATTR" />
28.     </xs:complexType>
29.
30.     <xs:attributeGroup name="SongTypeATTR">
31.         <xs:attribute name="song_id" type="xs:ID" use="required" />
32.         <xs:attribute name="Genre" type="xs:string" use="required" />
33.         <xs:attribute name="Length" type="xs:string" use="required" />
34.     </xs:attributeGroup>
```

```
35.
36.     <xs:complexType name="WritersType">
37.         <xs:sequence>
38.             <xs:group ref="WritersTypeGR" />
39.         </xs:sequence>
40.     </xs:complexType>
41.
42.     <xs:complexType name="ArtistsType">
43.         <xs:sequence>
44.             <xs:group ref="ArtistsTypeGR"></xs:group>
45.         </xs:sequence>
46.     </xs:complexType>
47.
48.     <xs:complexType name="AwardsType">
49.         <xs:sequence>
50.             <xs:group ref="AwardsTypeGR"></xs:group>
51.         </xs:sequence>
52.     </xs:complexType>
53.
54.     <xs:simpleType name="TelephoneST">
55.         <xs:restriction base="xs:positiveInteger">
56.             <xs:pattern value="023[0-9]{6}" />
57.         </xs:restriction>
58.     </xs:simpleType>
59.
60.     <xs:simpleType name="PhoneST">
61.         <xs:restriction base="xs:positiveInteger">
62.             <xs:pattern value="98[0-9]{8}" />
63.         </xs:restriction>
64.     </xs:simpleType>
65.
66.     <xs:simpleType name="writerST">
67.         <xs:restriction base="xs:string">
68.             <xs:enumeration value="Male" />
69.             <xs:enumeration value="Female" />
70.             <xs:enumeration value="Other" />
71.         </xs:restriction>
```

```
72.     </xs:simpleType>
73.
74.     <xs:simpleType name="s_genderST">
75.         <xs:restriction base="xs:string">
76.             <xs:enumeration value="Male" />
77.             <xs:enumeration value="Female" />
78.             <xs:enumeration value="Other" />
79.         </xs:restriction>
80.     </xs:simpleType>
81.
82.     <xs:simpleType name="d_genderST">
83.         <xs:restriction base="xs:string">
84.             <xs:enumeration value="Male" />
85.             <xs:enumeration value="Female" />
86.             <xs:enumeration value="Other" />
87.         </xs:restriction>
88.     </xs:simpleType>
89.
90.     <xs:simpleType name="g_genderST">
91.         <xs:restriction base="xs:string">
92.             <xs:enumeration value="Male" />
93.             <xs:enumeration value="Female" />
94.             <xs:enumeration value="Other" />
95.         </xs:restriction>
96.     </xs:simpleType>
97.
98.     <xs:group name="Music_StoreGR">
99.         <xs:sequence>
100.             <xs:element name="Name" type="xs:string" />
101.             <xs:element name="Address" type="xs:string" />
102.             <xs:element name="Contact" type="ContactType" />
103.             <xs:element name="Website_address" type="xs:string" />
104.             <xs:element name="Logo" type="xs:string" />
105.             <xs:element name="Songs" type="SongsType" />
106.         </xs:sequence>
107.     </xs:group>
108.
```

```
109.     <xs:group name="ContactTypeGR">
110.         <xs:sequence>
111.             <xs:element name="Telephone" type="TelephoneST" />
112.             <xs:element name="Phone" maxOccurs="unbounded" type="PhoneST" />
113.             <xs:element name="Facebook_Page" type="xs:string" />
114.         </xs:sequence>
115.     </xs:group>
116.     <xs:group name="SongsTypeGR">
117.         <xs:sequence>
118.             <xs:element name="Song" maxOccurs="unbounded" type="SongType" />
119.         </xs:sequence>
120.     </xs:group>
121.     <xs:group name="SongTypeGR">
122.         <xs:sequence>
123.             <xs:element name="Title" type="xs:string" />
124.             <xs:element name="Language" type="xs:string" />
125.             <xs:element name="Writers" type="WritersType" />
126.             <xs:element name="Artists" type="ArtistsType" />
127.             <xs:element name="Album" type="xs:string" />
128.             <xs:element name="Released" type="xs:string" />
129.             <xs:element name="Movie_Playback" />
130.             <xs:element name="Awards" type="AwardsType" />
131.             <xs:element name="Director" type="xs:string" />
132.             <xs:element name="Composer" type="xs:string"
maxOccurs="unbounded" />
133.             <xs:element name="Producer" type="xs:string" />
134.         </xs:sequence>
135.     </xs:group>
136.
137.     <xs:group name="WritersTypeGR">
138.         <xs:sequence>
139.             <xs:element name="writer" maxOccurs="unbounded">
140.                 <xs:complexType>
141.                     <xs:simpleContent>
142.                         <xs:extension base="xs:string">
143.                             <xs:attributeGroup ref="WritersTypeGRATTR" />
144.                         </xs:extension>
```

```

145.             </xs:simpleContent>
146.         </xs:complexType>
147.     </xs:element>
148. </xs:sequence>
149. </xs:group>
150. <xs:attributeGroup name="WritersTypeGRATTR">
151.     <xs:attribute name="wr_id" type="xs:ID" />
152.     <xs:attribute name="wr_gender" type="writerST" />
153. </xs:attributeGroup>
154. <xs:group name="ArtistsTypeGR">
155.     <xs:sequence>
156.         <xs:element name="Band" type="xs:string" minOccurs="1" />
157.         <xs:element name="Singers" minOccurs="1">
158.             <xs:complexType>
159.                 <xs:sequence>
160.                     <xs:element name="Solo" minOccurs="0">
161.                         <xs:complexType>
162.                             <xs:sequence>
163.                                 <xs:element name="S_Singer" />
164.                             </xs:sequence>
165.                             <xs:attributeGroup ref="ArtistsTypeGRATTR"
166.                             />
167.                         </xs:complexType>
168.                     </xs:element>
169.                     <xs:element name="Duets" minOccurs="0">
170.                         <xs:complexType>
171.                             <xs:sequence>
172.                                 <xs:element name="D_Singer"
173.                                 minOccurs="1" maxOccurs="unbounded" />
174.                             </xs:sequence>
175.                             <xs:attributeGroup ref="DuetsATTR" />
176.                         </xs:complexType>
177.                     </xs:element>
178.                     <xs:element name="Group" minOccurs="0">
179.                         <xs:complexType>
180.                             <xs:sequence>

```

```
179.                <xs:element name="G_Singer"
maxOccurs="unbounded" />
180.                </xs:sequence>
181.                <xs:attributeGroup ref="GroupATTR" />
182.                </xs:complexType>
183.            </xs:element>
184.        </xs:sequence>
185.    </xs:complexType>
186. </xs:element>
187. </xs:sequence>
188. </xs:group>
189. <xs:attributeGroup name="ArtistsTypeGRATTR">
190.     <xs:attribute name="ssing_id" type="xs:ID" />
191.     <xs:attribute name="s_gender" type="s_genderST" />
192. </xs:attributeGroup>
193.
194. <xs:attributeGroup name="DuetsATTR">
195.     <xs:attribute name="dsing_id" type="xs:ID" />
196.     <xs:attribute name="d_gender" type="d_genderST" />
197. </xs:attributeGroup>
198. <xs:attributeGroup name="GroupATTR">
199.     <xs:attribute name="gsing_id" type="xs:ID" />
200.     <xs:attribute name="g_gender" type="g_genderST" />
201. </xs:attributeGroup>
202. <xs:group name="AwardsTypeGR">
203.     <xs:sequence>
204.         <xs:element name="Award" type="xs:string" maxOccurs="unbounded"
/>
205.     </xs:sequence>
206. </xs:group>
207. </xs:schema>
208.
```


Report

Testing

Testing of any project is necessary to identify mistakes and correct it to get the best end result. Testing is done to identify risks and manage it. To be sure that system meets the stated objectives and give best service to end user, good testing is practiced. Below are the testing of xml project of 'Jethako Music Store'.

TEST 1: XML validation without using CSS

Test No.	1
Action	CSS was removed from XML file.
Expected Output	Whole XML file must appear in tree structure in browser.
Actual Output	Whole XML file appeared in tree structure in browser (chrome in my case) when CSS file was commented.
Test Result	Test successful.

Table 2 Test 1 testing table

```

<!-- <?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?> -->
<!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->
▼<Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">
  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  ▼<Contact>
    <Telephone>023590972</Telephone>
    <Phone>9814978419</Phone>
    <Phone>9816924453</Phone>
    <Facebook_Page>Jethako Pasal</Facebook_Page>
  </Contact>
  <Website_address>www.Jethakopasal.com</Website_address>
  <Logo/>
  <!-- Song 1 -->
  ▼<Songs>
    ▼<Song song_id="song_1" Genre="AlternativeIndie" Length="4:22">
      <Title>Tum Hi Ho</Title>
      <Language>Hindi</Language>
      ▼<Writers>
        <writer wr_id="wr_1" wr_gender="Male">Mithun Sharma</writer>
      </Writers>
      ▼<Artists>
        <Band/>
        ▼<Singers>
          ▼<Solo>
            <S_Singer sasing_id="sing_1" s_gender="Male">Arijit Singh</S_Singer>
          </Solo>
        </Singers>
      </Artists>
      <Album>Aashiqui 2</Album>
      <Released>16 March 2013</Released>
      <Movie_Playback>Aashiqui 2</Movie_Playback>
      ▼<Awards>
        <Award>Filmfare Award for Best Male Playback Singer</Award>
        <Award>Best Music Director</Award>
        <Award>Best Lyrics 2014</Award>
        <Award>Most Entertaining Song</Award>
        <Award>Song of the year(2014)</Award>
        <Award>6th Mirchi Music Awards</Award>
      </Awards>
      <Director>Mithun Sharma</Director>
      <Composer>Mithun Sharma</Composer>
      <Producer>Mukesh Bhatt and Team</Producer>
    </Song>
  </Songs>
</Music_Store>

```

Figure 2 Test 1

TEST 2: XML validation with DTD

Test No.	2
Action	XML and DTD file was uploaded one after another in xmlvalidation.com for validation
Expected Output	No errors must come
Actual Output	No errors was found.
Test Result	Test successful

Table 3 Test2 testing table.

Please copy your XML document in here:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- <?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?
> -->
<!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd">

<!-- <Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd"> -->
<Music_Store>
  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
```

Or upload it:

Choose File

No file chosen

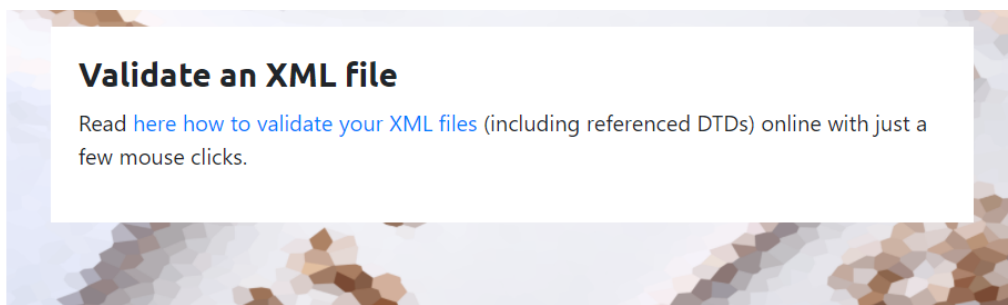
The validation check is performed against any XML schema or DTD declared inside the XML document.

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

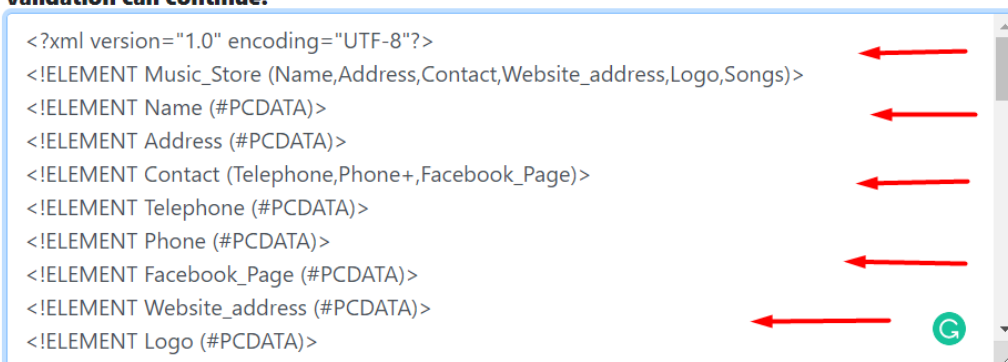
To validate the XML document against an external XML schema, click below.

☐ Validate against external XML schema

Figure 3 Test2. Uploading XML file in xmlvalidation.com



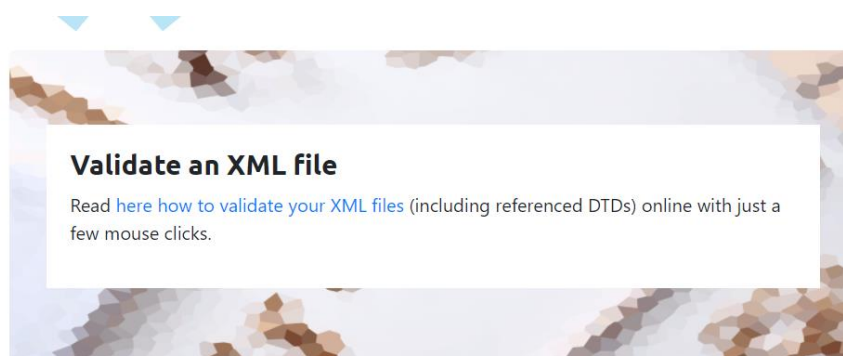
The file catalog_19031860.dtd is being referenced. Please copy it in here, so that the validation can continue:



Or upload it:

No file chosen

Figure 4 Test2. Uploading DTD file to xmlvalidation.com as required.



No errors were found

The following files have been uploaded so far:

[XML document:](#)

[catalog_19031860.dtd](#)

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

Figure 5 Test2. No errors found in above XML validation with DTD.

TEST 3: XML validation with Schema

Test No.	3
Action	First XML file was uploaded in xmlvalidation.com followed by XSD file for validation.
Expected Output	No errors must arise.
Actual Output	No errors was found.
Test Result	Test successful by meeting expected result.

Table 4 Test 3 testing table.

Please copy your XML document in here:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- <?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?
> -->
<!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->

<Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">
  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  <Contact>
```

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 6 Test3. Uploading XML file in xmlvalidation.com

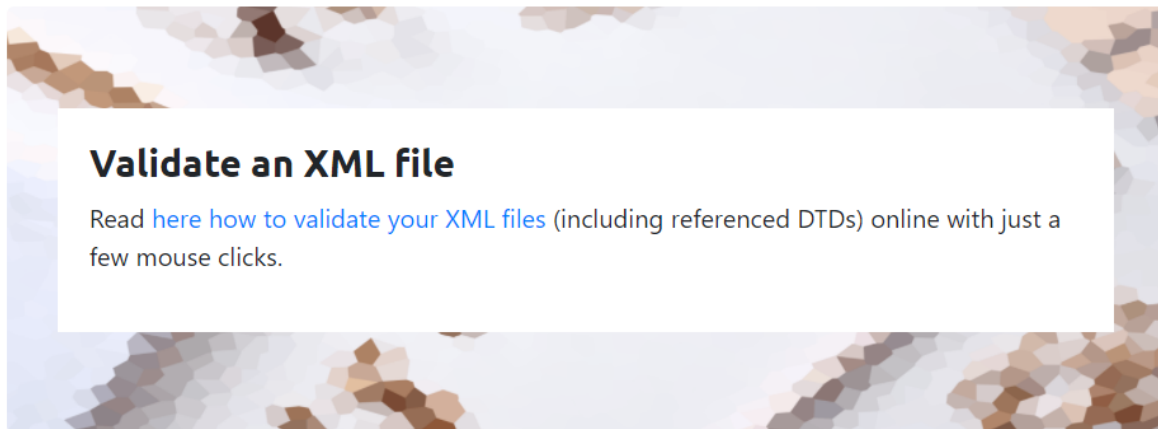
The file catalog_19031860.xsd is being referenced. Please copy it in here, so that the validation can continue:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Music_Store">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Name" type="xs:string" />
        <xs:element name="Address" type="xs:string" />
        <xs:element name="Contact">
          <xs:complexType>
            <xs:sequence>
```

Or upload it:

No file chosen

Figure 7 Test3. Uploading XSD file in xmlvalidation.com as requested.



No errors were found

The following files have been uploaded so far:

XML document:

[catalog_19031860.xsd](#)

[catalog_19031860.dtd](#)

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

Figure 8 Test3. No errors found in XML validation with XSD

TEST 4: XML validation with CSS

Test No.	4
Action	First load XML file without CSS. Then load it with combining CSS.
Expected Output	XML file must load in browser with beautiful design.
Actual Output	XML file was loaded in browser with beautiful design.
Test Result	Test successful with expected output.

Table 5 Test4. XML with and without CSS.

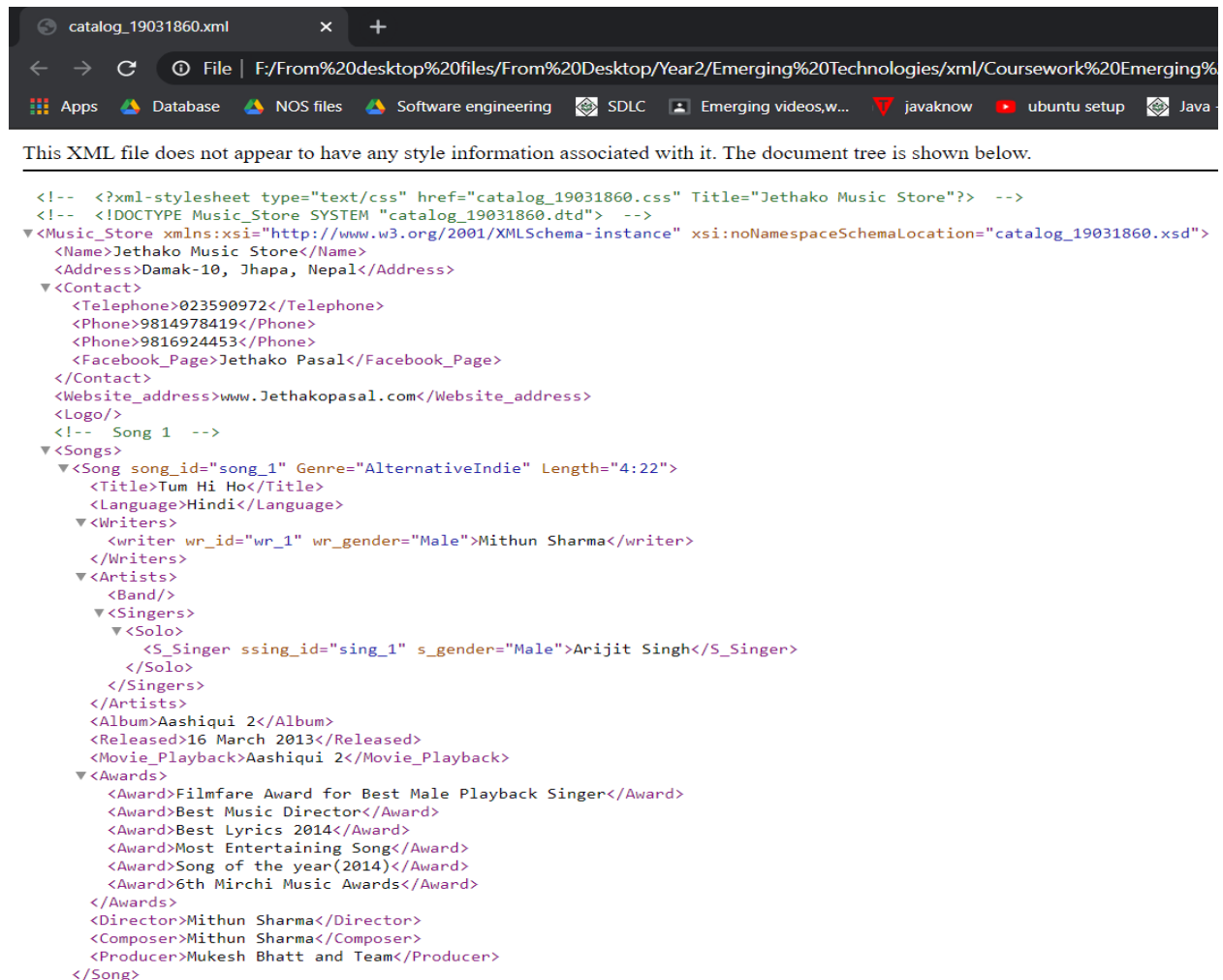


Figure 9 Test4. XML file without CSS.



Figure 10 Test4. XML file with CSS

TEST 5: XML validation with DTD for Modifying symbols.

Test No.	5
Action	First removing modified element from XML document to validate it with DTD. Proceed validation using xmlvalidation.com
Expected Output	Must show error in the absence of modified element
Actual Output	Error was shown that it must match the modified element, where (writer+) is the case above in my validation.
Test Result	Test successful after meeting expected output.

Table 6 Test 5 testing XML modification using Schema

```

<!ELEMENT Song (Title,Language,Writers,Artists,Album,Released,Movie_Playback,Awards,Director*,Composer+,Producer*)>
<!ATTLIST Song song_id ID #REQUIRED>
<!ATTLIST Song Genre CDATA #REQUIRED>
<!ATTLIST Song Length CDATA #REQUIRED>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Language (#PCDATA)>
<!ELEMENT Writers (writer+) <--
<!ELEMENT writer (#PCDATA)>
<!ATTLIST writer wr_id ID #REQUIRED>
<!ATTLIST writer wr_gender (Male | Female) #REQUIRED>

```

Figure 11 Test5 XML validation of Modifying Symbol test with DTD

Please copy your XML document in here:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
<!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd">



<!-- <Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd"> -->
<Music_Store>
  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  <Contact>
```

Or upload it:



No file chosen

Figure 12 Test5. Uploading XML document in xmlvalidation.com

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:

 26: 23 The content of element type "Writers" is incomplete, it must match "(writer)+". 

XML document:

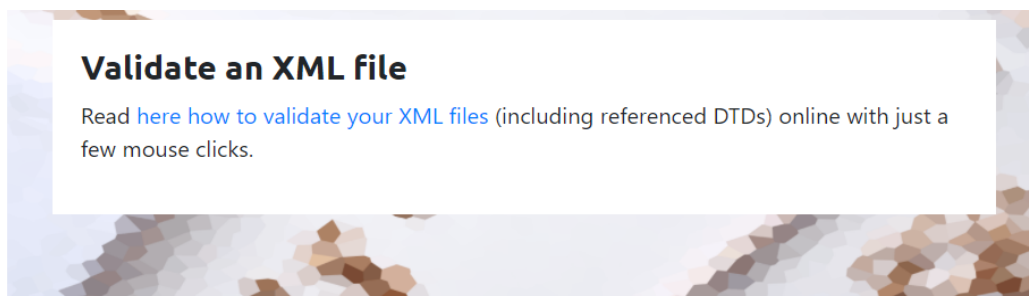
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
3 <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd">
4
5 <!-- <Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
6 xsi:noNamespaceSchemaLocation="catalog_19031860.xsd"> -->
7 <Music_Store>
```

Figure 13 Test5. Uploading DTD Document for validation.

TEST 6: XML validation with Schema for indicators

Test No.	6
Action	First repeating element that was supposed not to be repeated as per Schema.
Expected Output	Error was supposed to be shown.
Actual Output	Error was shown when it was validated in xmlvalidation.com
Test Result	Test successful by meeting expected output.

Table 7 Test6 validating indicator of xml using Schema



Please copy your XML document in here:



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
<!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd" -->

<Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">



  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  <Contact>
```

Figure 14 Test6. XML validation for testing indicator using Schema

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:

- ✖ 99: 28 cvc-complex-type.2.4.a: Invalid content was found starting with element 'Duets'. One of
'{"":Group}' is expected. 
- ✖ 101: 29 cvc-complex-type.2.4.b: The content of element 'Duets' is not complete. One of
'{"":D_Singer}' is expected. 

XML document:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
3 <!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->
4

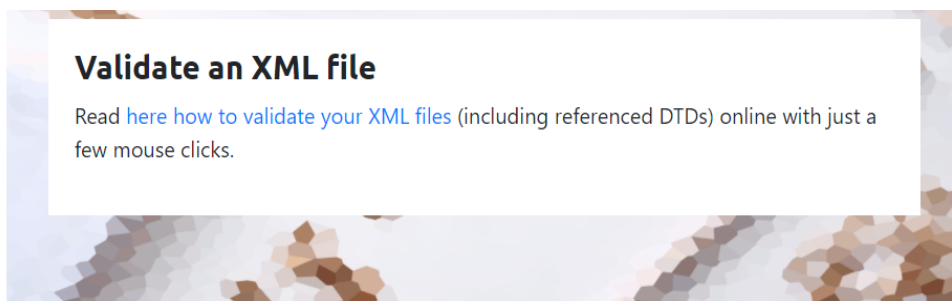
```

Figure 15 Test6. Error shown by Schema for breaking the rule of indicator minoccurs

TEST 7: XML validation with Schema for data type

Test No.	7
Action	Instead of integer type string value was inserted in Phone element of XML
Expected Output	Error message that string is not allowed in integer data type
Actual Output	Error message was displayed.
Test Result	Test successful by meeting expected output.

Table 8 Test7 validating data type of XML using Schema



Please copy your XML document in here:



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
<!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->

<Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">



  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  <Contact>
```

Figure 16 Test7. Validating XML for data type using Schema.

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:

-  12: 54 cvc-pattern-valid: Value 'phonenumber should be positive' is not facet-valid with respect to pattern '98[0-9]{8}' for type 'null'.
-  12: 54 cvc-type.3.1.3: The value 'phonenumber should be positive' of element 'Phone' is not valid.

XML document:

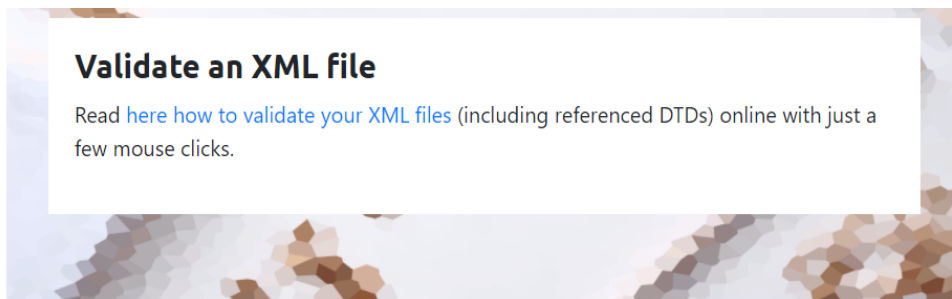
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
3 <!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->
4
5 <Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

Figure 17 Test7. Error display for data type in XML using Schema

TEST 8: XML validation with Schema for Restriction of Gender

Test No.	8
Action	Writing another word rather than the genders already defined by Schema
Expected Output	Error must arise telling the entered word is not valid.
Actual Output	Error was displayed by Schema that we are not allowed to write gender other than the ones defined in Schema.
Test Result	Test successful by meeting expected output.

Table 9 Test8. Testing restriction if XML using Schema



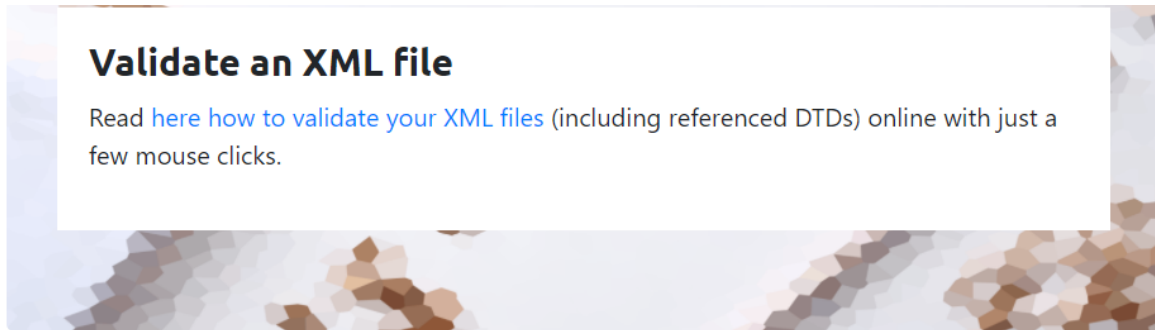
Please copy your XML document in here:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
<!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->

<Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">

  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  <Contact>
```

Figure 18 Test8. Validating XML using Schema for restriction

**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:

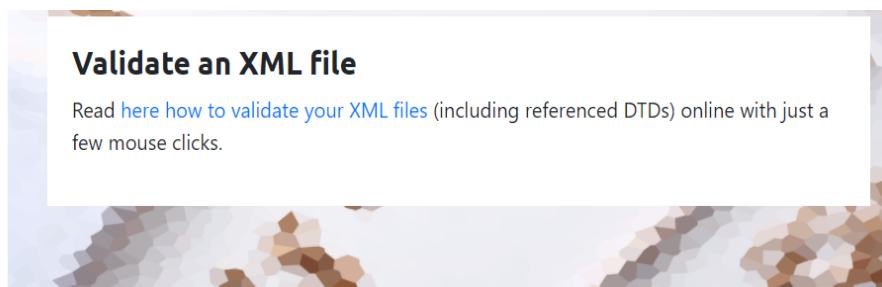
- ✖ 25: 58 cvc-attribute.3: The value 'Nothing' of attribute 'wr_gender' on element 'writer' is not valid with respect to its type, 'null'.
- ✖ 25: 58 cvc-enumeration-valid: Value 'Nothing' is not facet-valid with respect to enumeration '[Male, Female, Other]'. It must be a value from the enumeration. ←

Figure 19 Test8. Error displayed for restriction.

TEST 9: XML validation with Schema for Restriction of Telephone number

Test No.	9
Action	Entering random number in Telephone
Expected Output	Error in XML validation using Schema
Actual Output	Error was developed in validation telling number must start from 023
Test Result	Test was successful meeting expected output.

Table 10 Test9. Testing telephone restriction of XML



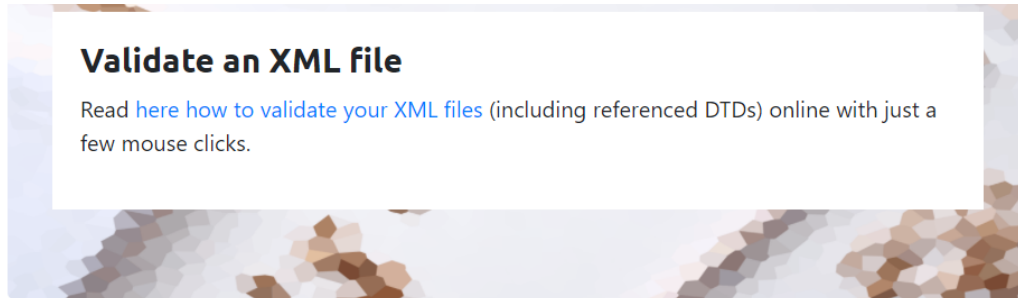
Please copy your XML document in here:



```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
<!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd"> -->

<Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">

  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  <Contact>
```

Figure 20 Test9. Validating Telephone element of XML

**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:



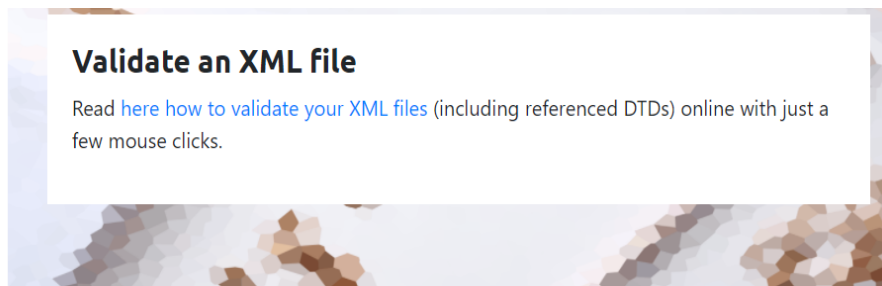
-  11:41 cvc-pattern-valid: Value '977590972' is not facet-valid with respect to pattern '023[0-9]{6}' for type 'null'.
-  11:41 cvc-type.3.1.3: The value '977590972' of element 'Telephone' is not valid.

Figure 21 Test9. Error by Schema for Telephone

TEST 10: XML validation with Schema for Unique ID attribute.

Test No.	10
Action	Writing same ID for song and writer
Expected Output	Error must be displayed.
Actual Output	Error was displayed, id of song and writer cannot be same.
Test Result	Test was successful meeting expected output.

Table 11 Test 10 validating unique Id of XML Using Schema

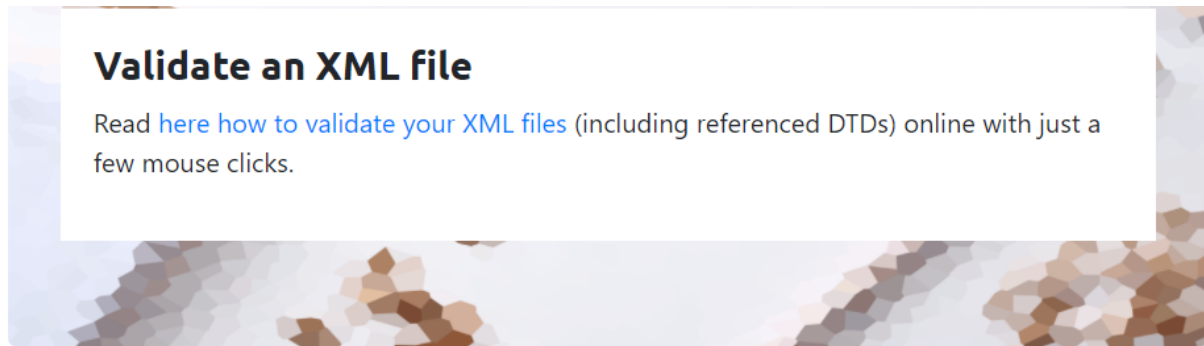
Please copy your XML document in here:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="catalog_19031860.css" Title="Jethako Music Store"?>
<!-- <!DOCTYPE Music_Store SYSTEM "catalog_19031860.dtd" -->

<Music_Store xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="catalog_19031860.xsd">

  <Name>Jethako Music Store</Name>
  <Address>Damak-10, Jhapa, Nepal</Address>
  <Contact>
```

Figure 22 Test10. XML validation for Unique ID using Schema



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:

- ✖ 25: 57 cvc-attribute.3: The value 'song_1' of attribute 'wr_id' on element 'writer' is not valid with respect to its type, 'ID'.
- ✖ 25: 57 cvc-id.2: There are multiple occurrences of ID value 'song_1'.

XML document:

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

Figure 23 Test 10. Error displayed due to same ID

Difference between Schema and DTD

Let us know differences in points then read it in table for easiness.

1. Name and extension

DTD stands for 'Document Type Definition' with dtd file extension where as XSD stands for 'XML Schema Definition' with xsd file extension. (knowprogramXSDvsDTD, 2021)

2. Derived

DTD syntax are derived from SGML (Standard Generalized markup language), whereas XSD uses XML-based syntax. (JavaTpointDTDVSXSD, 2021)

3. Datatypes

DTD doesn't contain real datatype but has #PCDATA as string datatype for elements whereas XSD has primitive/fundamental and Custom data types (which contain child elements or attributes) such as: string, decimal, float, Boolean, etc.

4. Order:

DTD allows us to define child element and their attributes in any order whereas XSD sets rule that child elements, their attributes in sequence which is specific order. (JavaTpointDTDVSXSD, 2021)

5. Extensibility:

DTD does not allow to make new tags whereas XSD allow to make new tags like XML.

6. Namespace:

We cannot use namespace privilege in DTD because it has its own set of keywords for defining a schema, for example:

!DOCTYPE for a root tag.

!ELEMENT for an element

!ATTLIST for an attribute

!ENTITY for defining variables.

Whereas in XSD we can define our own set of namespaces. (JavaTpointDTDVSXSD, 2021)

7. Restrictions

No restrictions are in DTD but XSD allows us to specify restriction.

8. Checking

DTD is weakly typed because It checks only grammar and validity of a XML document where as DTD is strong typed because It also checks the correctness of the structure of the XML so they don't require parser.

9. Use

DTD is used especially for small XML files but XSD is used for large XML files like web service because Schema is really powerful

10. Definition:

DTD supports inline definition which means we can define DTD inside an XML document but XSD cannot be defined in instance document.

11. Control:

DTD has less control over XML compared to XSD because it does not define in detail to attributes and elements in XML

12. Reusability

DTD poorly supports code reuse and can use parameter entities whereas XSD can reuse definitions using named typed. For example, we can use GenderType again and again to define any gender element for datatype. (JavaTpointDTDVSXSD, 2021)

Below table is a short revise of above explained points.

S.N	DTD	XSD
1	It is Document Type Definition with dtd as file extension.	It is SML Schema Definition with xsd as file extension.
2.	DTDs are derived from SGML	XSDs are written in XML.
3	It doesn't support data types.	It supports different datatypes.
4	It doesn't maintain order for elements.	It maintains order.
5	It is not extensible.	It is extensible.
6	It does not support namespace.	Supports namespace for defining schema.
7	Absence of restrictions	Restriction privilege is present.
8	Especially used for small XML files.	Used for large XML files.
9	It is weakly typed.	It is strongly typed.
10	Allows inline definitions.	Doesn't allow inline definition.
11	Has less control over XML.	Has more control over XML.
12	Poorly supports code reuse	Strongly allows code reuse.

Table 12 Difference between DTD and Schema

The above table is referenced from (JavaTpointDTDVSXSD, 2021).

How the coursework was developed?

In short, coursework was developed with my blood, sweat, and pain. Apart from jokes, I would like to classify my journey of development into the following phase:

Research

This part here is the brain of this CW. The research was made on XML, DTD, XSD, their differences, testing, few XSD design patterns, CSS, CSS beautiful designs, and many more. The research that made me burn out was: Songs and its writer, producer, director, composer, singer, genre, length, nominations, movie name, and release date.

Coding

This part here is the soul of CW. The CW demanded to develop a system of music store using XML. The XML was combined with CSS and use DTD and XML Schema for validation. The tree diagram was the foundation of coding. It was not easy to draw it, thanks to our module teacher, who continuously corrected me to execute the best tree structure. After creating it, XML was written which was easy. To validate it, DTD was written and combined. Writing XSD was a tough challenge for me. For XSD, the Venetian Blind design pattern is used, which I have explained in the appendix section.

Tools

Tools are the organs of this CW. Tools that I used to develop this whole CW are:

✓ **draw.io:**

This software was designed by Seibert Media which I used to create tree diagram. Though its prices start at \$5 per month for ten users, and \$577.50 for 5000 users but I downloaded it for free from Microsoft store and used it. It is used for making diagrams and charts. It allows us to choose from an automatic layout function, or create a custom layout. It collects about hundreds of shapes and visual effect. The drop-and -drag feature makes it simple too create a great looking diagram or chart. (Computerhopedraw.io, 2021)



Figure 24 Draw.io tool

✓ **Visual Studio Code (VSC):**

It is a light weight code editor which runs on desktop with the support of OS like Windows, mac and Linux. It has a built-in support for JavaScript, TypeScript and Node.js with the ecosystem of many other languages. VSC is powerful tool because it provides developer tooling, like IntelliSense code completion and debugging. I used it to create my XML, DTD, XSD, and XML files. (VSCdocumentation/docs, 2021)

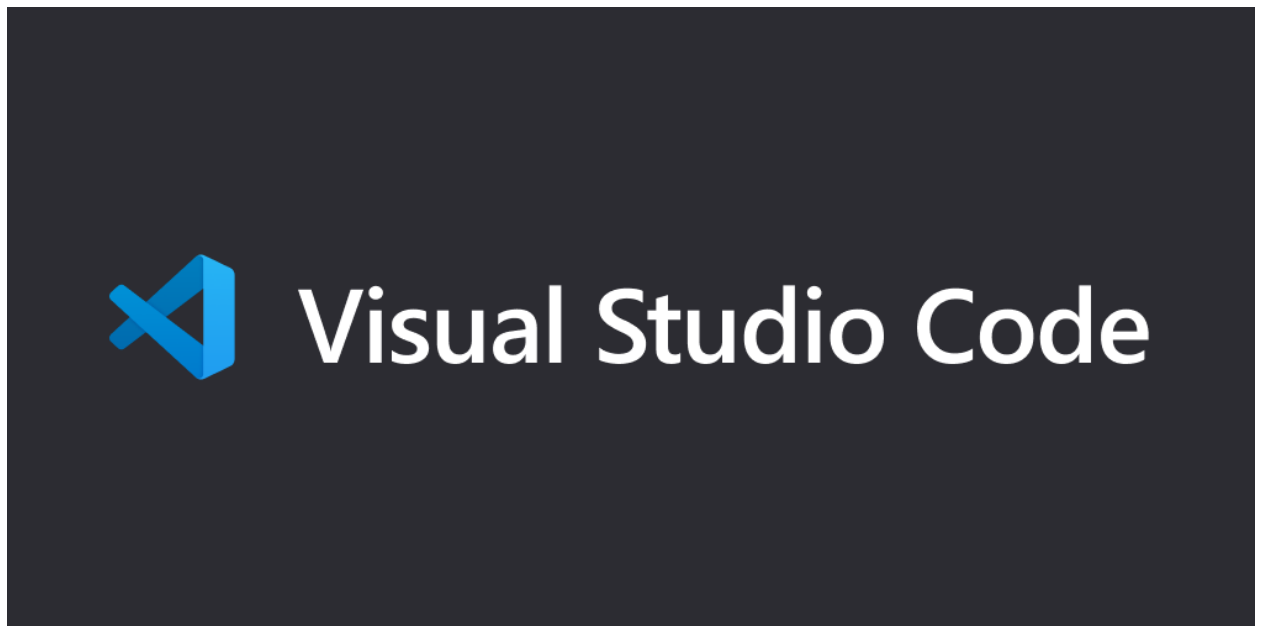


Figure 25 Visual Studio code tool.

✓ **xmlvalidation.com:**

It is a online XML validating website provided by Theano GmbH. I used it continuously to validate my XML file with DTD and XSD.



Figure 26 xmlvalidation.com tool.

✓ **Chrome:**

We all know that google chrome is a cross-platform web browser, officially released by google on December 11, 2008. It was used to research the above research topics which I mentioned. With its help only I was validating my files. (computerhopeChrome, 2021)



Figure 27 chrome tool

✓ **Lightshot:**

It is a free software and helps to take a customizable screenshot. to click it we should have to make ctrl + prtSc key combination, select the area and wither save or copy screenshot to clipborad. It was used to take screenshots for testing. (Lightshot, 2021)



Figure 28 KightShot tool.

✓ **Microsoft word:**

It belongs to MS office but also can be bought separately. It was first released in 1981 and till now it's a famous graphical word processing software. I used it for documentation. (computerhopeMSWORD, 2021)



Figure 29 MSWord tool.

✓ **Easy Code formatter:**

It is a free software used for documentation or to format code by providing us multiple coding themes, line rendering option and highlighting. It was used to convert/format text sentences into code-like structures in Word.



Figure 30 Easycodeformatter

✓ **Prepostseo:**

It is a free search engine optimization company developed by two Pakistani brothers in Ahmad Satar. They contain best SEO solutions. Their best tool are plagiarism checker and online Ping website tool. I used their plagiarism checker to check my documentation plags. (Prepostseo/aboutus, 2021)

🌈 **Documentation**

This part here is the soul of CW. It includes the development and documentation phases of CW. Whole texts are written in Arial font with font size 12. Proper header and footer were given which are module code, module name and student name, UID respectively. The whole documentation is thoroughly checked through plagiarism and grammar checker with perfect word count which is 1500.

Critical Analysis

Coursework plays a great role for students to revise what they have studied the whole semester. Not only to recall the lessons but also it enhances the skill of research and doing work on own, which boosts students' confidence and knowledge. During the whole journey, some of the life lessons that I have learned and want to reflect on are:

1.Procrastination:

I could have finished coursework many days ago but I procrastinated my work which led me to rush in the last few days of submission. We should always remember that slow and steady wins the race.

2.Begin from the beginning.

After I started to worry and rush, I directly started to do coursework without learning the basics. I encourage everyone to start from basics before doing anything big or small.

3.Patience:

Patience is not the ability to wait, but the ability to keep a good attitude while waiting. We students should make patience our best friend. It will always help us in the time of trouble, especially while doing this type of CW.

4.Learn little by little.

It takes a step at a time to reach the top. If I could have learned CSS or photoshop little by little a few years or months back when I had sacks of free time then it could have opened many doors of opportunity which includes this coursework too. But careless me.

5.Better ask help (Help me!):

We should never hesitate and step back to ask for help or raise a question about what we haven't understood to teacher or friend. I didn't ask for help from anyone which made me suffer bitterly. If requesting help removes your iceberg then we should better do it.

6.A friend in need is a friend indeed:

I found a true friend in the midst of this coursework. When I was lost, troubled, and weary, I asked a friend to help, and surprisingly he helped me without hesitation. Wow, I am so lucky.

7.Learn from your mistakes.

I am of today is a corrected version of yesterday. I have made lots of scars, mistakes, and suffered failure which I count as a chapter of my life book which I study as a lesson. The above mistakes go to the book too, which I will never ever repeat. This is how we must grow and chase our dream

I take the above mistakes and problems as a teaser of what a programmer has to go through in his coding career. Problems led me to a solution, solution leads me to good marks and good marks lead me to success. If we get everything in life or if we can build anything with coding then there's no point in living and becoming a programmer because everyone could be built and be a programmer. I am very happy that the above problem arose. Above bitter experiences will make me a sweet programmer and lead my career with ease. Experiences and Lesson that I got, it will help me to move with turtle pace every day to acquire knowledge, get better every day and be the best programmer that I can be. Hope this is the same for my other friends and be successful.

Conclusion

In life we see, even long roads have an end to them, and the same goes for this coursework (CW). Finally, to conclude, I've learned about XML, CSS, Schema, DTD, and life lessons during the journey of CW. Their use in the IT world. After finishing the CW I had a blast of happiness and little sadness at the same time. I have learned so many things, especially from research and my mistakes. The CW itself was not that much difficult but things started to show up while doing Schema.

A lot of time I needed to ask the module teacher about my CW progress, but we all are suffering from the covid19 pandemic due to which studying from home sometimes can be difficult. But my basketful thanks goes to Mr. Pradhumna Dhungana sir for always being there and helping me when I was in trouble.

Learning new things is never easy, if it was easy then everyone would have studied IT and done it. I also suffered pain and depression when I was not able to pull out a good format for XML in a tree diagram. Sometimes I was not able to find out good research material. But after all, all that we can do is remain cool, patient, and keep hustling, and that's what I did. Due to which writing this conclusion today is possible.

Everything considered, in the end, I would like to say that becoming few days of XML developer by doing this CW is in some way fun and rewarding. I now have learned to sit in front of a laptop screen for several hours without distraction and complain. Hope to study in this same pace to be a good software engineer someday and make an impact in others life.

References

- Allaboutxml.* (2021, May 5). Retrieved from All About XML:
<https://xmlanditsusesandsyntax.blogspot.com/?view=classic#!>
- BriefHistoryOfXML.* (2021, May 6). Retrieved from Breif History: History of XML
Extensible markup language (XML) is used to format W3C pages. XML is a simplified form of SGML which was developed by Charles Goldfarb, Ed Mosher and Ray Lorie in the late 1970 while working at IBM. And on 10th Feb 1998, the WWW (W3C) publi
- computerhopeChrome.* (2021, May 7). Retrieved from Computer Hope:
<https://www.computerhope.com/jargon/c/chrome.htm>
- Computerhopedraw.io.* (2021, May 7). Retrieved from Computer hope:
<https://www.computerhope.com/jargon/d/drawio.htm>
- computerhopeMSWORD.* (2021, May 7). Retrieved from Computer Hope:
<https://www.computerhope.com/jargon/m/microsoft-word.htm>
- geeksforgeeksDTD.* (2021, May 5). Retrieved from GeeksforGeeks:
<https://www.geeksforgeeks.org/tag/web-technologies-html-and-xml/>
- JavaTpointDTDVSXSD.* (2021, May 7). Retrieved from JavaTpoint:
<https://www.javatpoint.com/dtd-vs-xsd>
- knowprogramXSDvsDTD.* (2021, May 6). Retrieved from KnowProgram:
<https://www.knowprogram.com/xml/xsd-vs-dtd/>
- Lightshot.* (2021, May 7). Retrieved from LightShot:
<https://app.prntscr.com/en/index.html#:~:text=Our%20app%20allows%20you%20to,screenshot%20with%202%20button%2Dclicks.&text=You%20can%20edit%20screenshots%20instantly,and%20find%20dozens%20similar%20images.>
- Prepostseo/aboutus.* (2021, May 7). Retrieved from Prepostseo:
<https://www.prepostseo.com/about>

VSCdocumentation/docs. (2021, May 7). Retrieved from Visual Studio Code:
<https://code.visualstudio.com/docs>

XMLDesignPattern. (2021, May 6). Retrieved from Oracle:
<https://www.oracle.com/technical-resources/articles/java/design-patterns.html>

Appendix (Optional):

Venetian Blind:

I have chosen a Venetian Blind design pattern for XSD. It's the same as Russian Doll and contains only one global element and other elements are local. It contains one root element. It helps the coder since it allows reuse for all the types and the single global element. Its only disadvantage is it limits encapsulation by exposing types. (XMLDesignPattern, 2021)

History of XML

Extensible markup language (XML) is used to format W3C pages. XML is a simplified form of SGML which was developed by Charles Goldfarb, Ed Mosher, and Ray Lorie in the late 1970 while working at IBM. And on 10th Feb 1998, the WWW (W3C) published XML 1.0 as a Recommendation. XML has become the most used and necessary tool for web services everywhere in the world. (BriefHistoryOfXML, 2021)

Should I learn XML?

Well, we should learn it only when we need it. We know XML formats the data for transferring, processing and storing. But lets us know the field where it is used and exists. It is use day-to-day data transactions such as :

Medical data

Financial transactions

Mathematical data

New information

Weather services etc.

There are several jobs in job portals with the best salaries, so we can take XML developer also as a career.