Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

DECLARATION

I understand that this is an individual assessment and that collaboration is not permitted. I have not received any assistance with my work for this assessment. Where I have used the published work of others, I have indicated this with appropriate citation.

I have not and will not share any part of my work on this assessment, directly or indirectly, with any other student.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at http://www.tcd.ie/calendar.

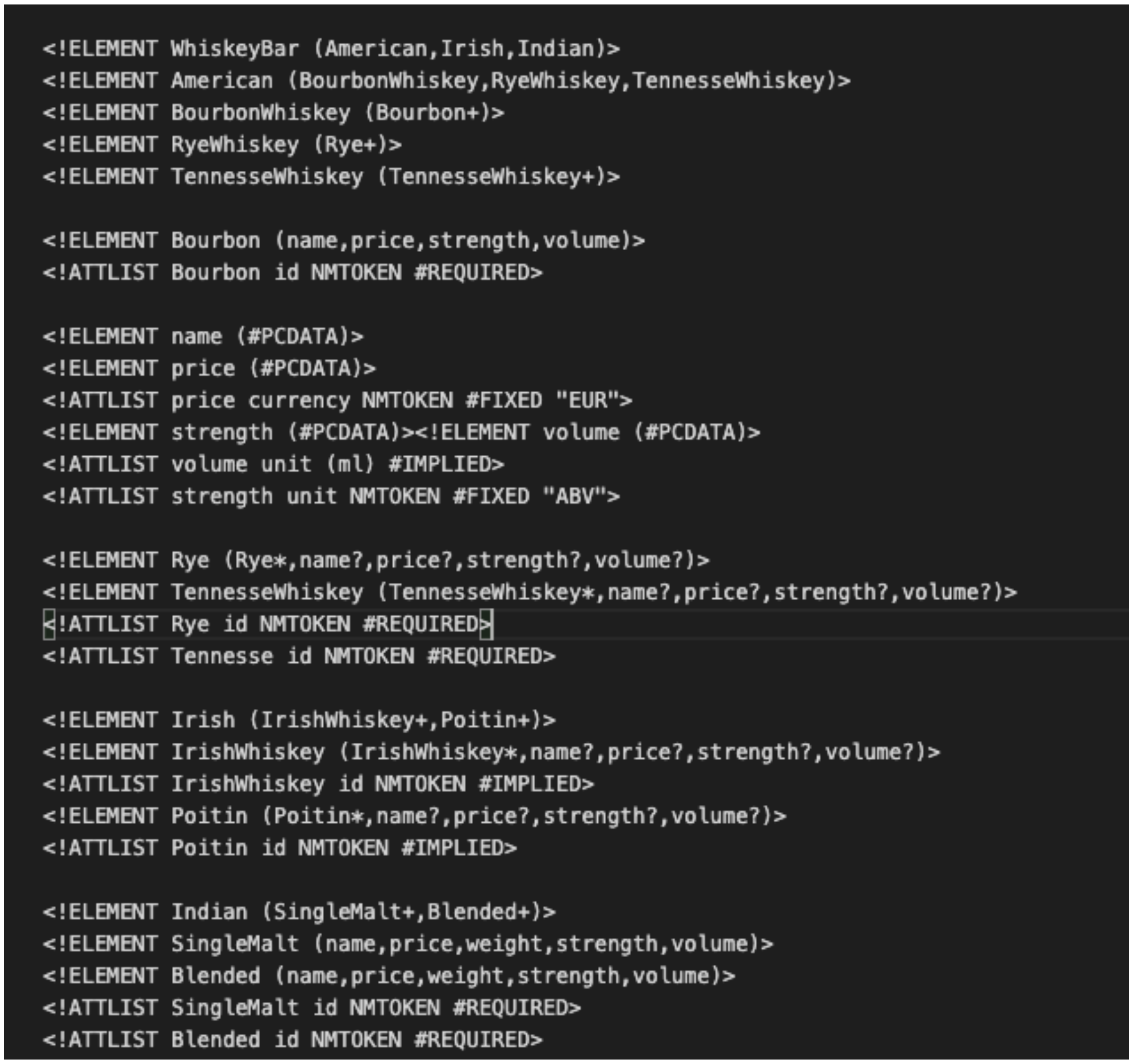
I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at http://tcd-ie.libguides.com/plagiarism/ready-steady-write."

I understand that by returning this declaration with my work, I am agreeing with the above statement. ✓

Name: Anirudh Kundu

Date: 11th August 2022

1. This DTD represents a Whiskey Bar, where it consists of American, Irish and Indian Whiskey, which are then individually further classified into their substituent types of Whiskey based on how they are labelled commercially (Single Malt, Bourbon Whiskey, etc.) . Realistically whiskeys are classified on a range of different attributes, but for the sake of generating a DTD which can further be implemented in xml and precisely scaled, a simple DTD of different attributes of Whiskey is as below:

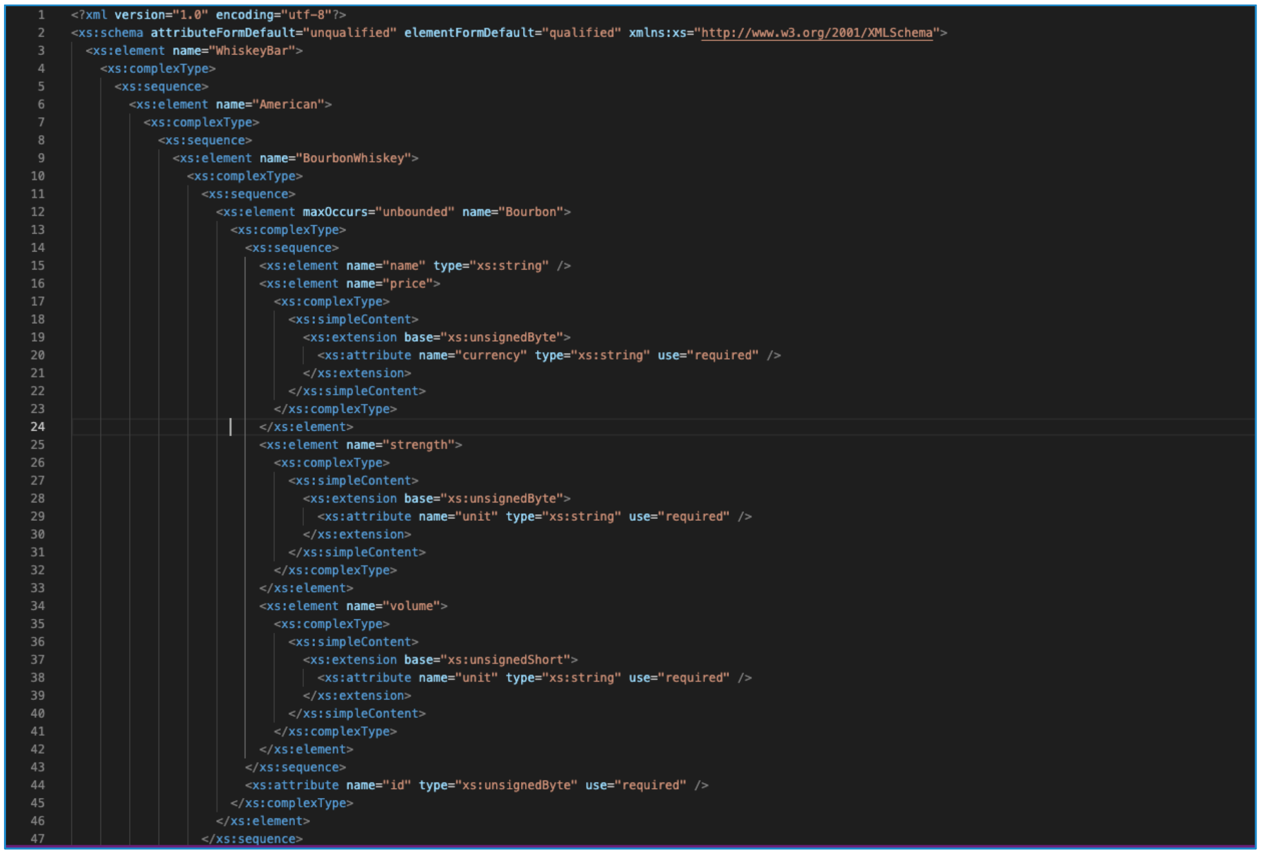


```
<!ELEMENT WhiskeyBar (American,Irish,Indian)>
<!ELEMENT American (BourbonWhiskey,RyeWhiskey,TennesseWhiskey)>
<!ELEMENT BourbonWhiskey (Bourbon+)>
<!ELEMENT RyeWhiskey (Rye+)>
<!ELEMENT TennesseWhiskey (TennesseWhiskey+)>

<!ELEMENT Bourbon (name,price,strength,volume)>
<!ATTLIST Bourbon id NMTOKEN #REQUIRED>

<!ELEMENT name (#PCDATA)>
<!ELEMENT price (#PCDATA)>
<!ATTLIST price currency NMTOKEN #FIXED "EUR">
<!ELEMENT strength (#PCDATA)><!ELEMENT volume (#PCDATA)>
<!ATTLIST volume unit (ml) #IMPLIED>
<!ATTLIST strength unit NMTOKEN #FIXED "ABV">

<!ELEMENT Rye (Rye*,name?,price?,strength?,volume?)>
<!ELEMENT TennesseWhiskey (TennesseWhiskey*,name?,price?,strength?,volume?)>
<!ATTLIST Rye id NMTOKEN #REQUIRED>
<!ATTLIST Tennesse id NMTOKEN #REQUIRED>

<!ELEMENT Irish (IrishWhiskey+,Poitin+)>
<!ELEMENT IrishWhiskey (IrishWhiskey*,name?,price?,strength?,volume?)>
<!ATTLIST IrishWhiskey id NMTOKEN #IMPLIED>
<!ELEMENT Poitin (Poitin*,name?,price?,strength?,volume?)>
<!ATTLIST Poitin id NMTOKEN #IMPLIED>

<!ELEMENT Indian (SingleMalt+,Blended+)>
<!ELEMENT SingleMalt (name,price,weight,strength,volume)>
<!ELEMENT Blended (name,price,weight,strength,volume)>
<!ATTLIST SingleMalt id NMTOKEN #REQUIRED>
<!ATTLIST Blended id NMTOKEN #REQUIRED>
```

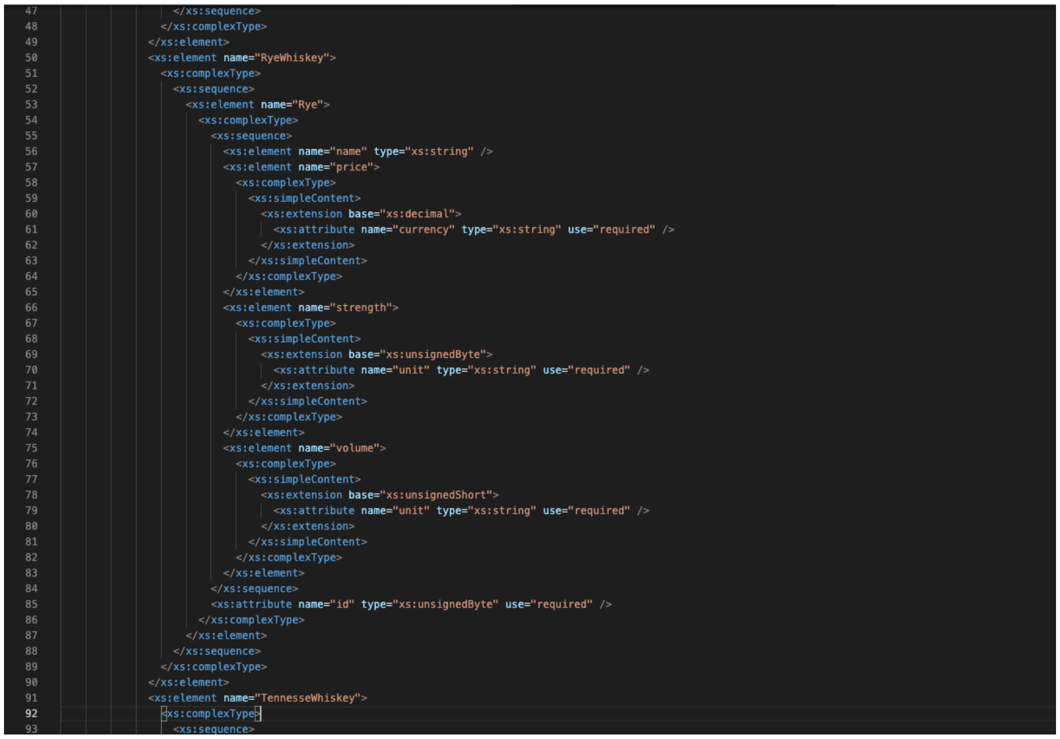
All Whiskey’s have been given a name token and depending on the nature of the data used, units such as ‘ml’, ‘ABV’ for alcohol concentration, are used. Units are marked as #REQUIRED, which means that any whiskey must have a minimum number of attributes, so as to be placed on the graph or the tree in this case.

2. XML mainly focuses on maintaining triples with various complexities, XSD however only gives a schema definition. XSD gives a clear hierarchy of the vocabulary as defined in the DTD, and clearly shows if an object* is bounded/unbounded (can have fixed/many numbers of values).

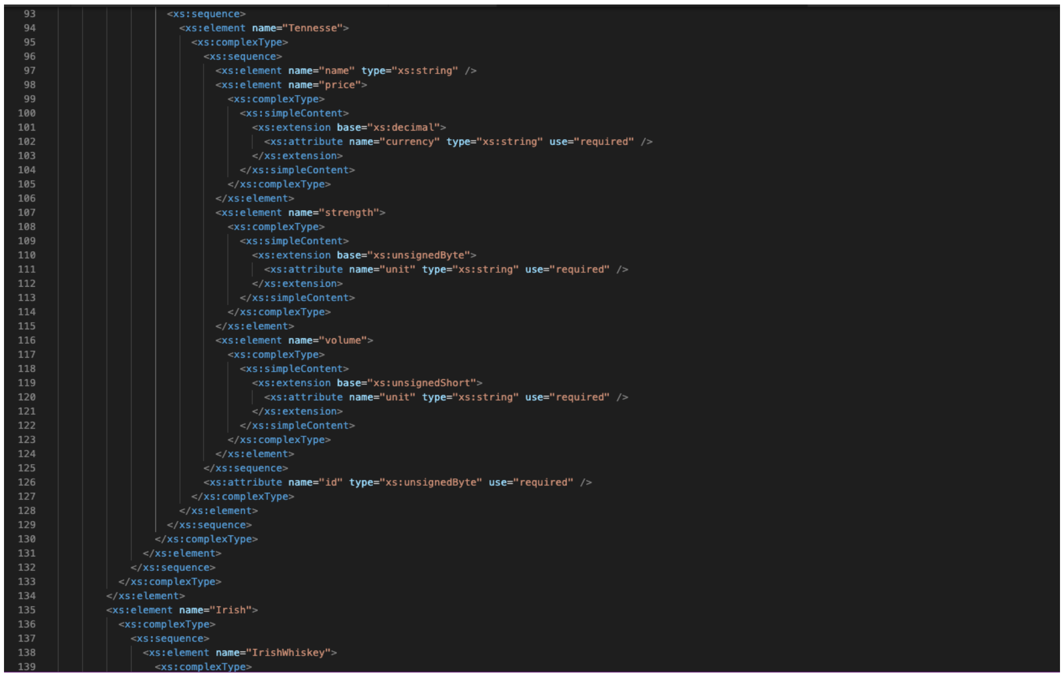
Thus, in this vocabulary some objects need to exist so that the vocabulary can be made into a valid XML.



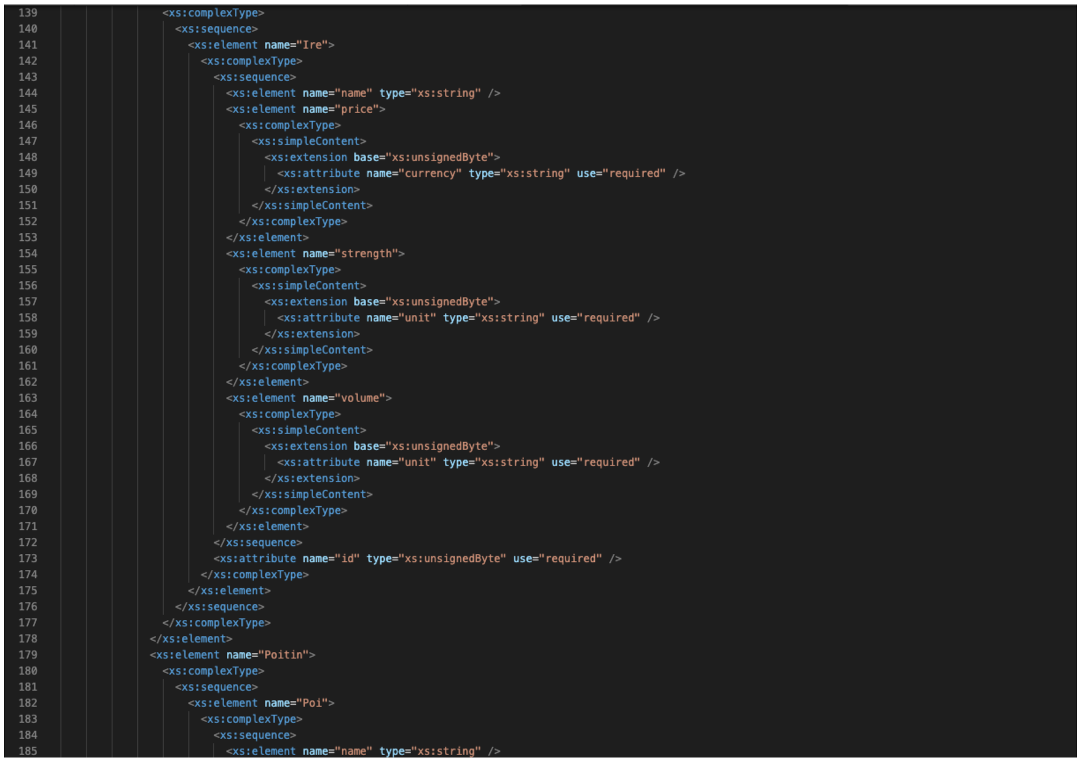
```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="WhiskeyBar">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="American">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="BourbonWhiskey">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element maxOccurs="unbounded" name="Bourbon">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="name" type="xs:string" />
                          <xs:element name="price">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="currency" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="strength">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="volume">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedShort">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:unsignedByte" use="required" />
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
```



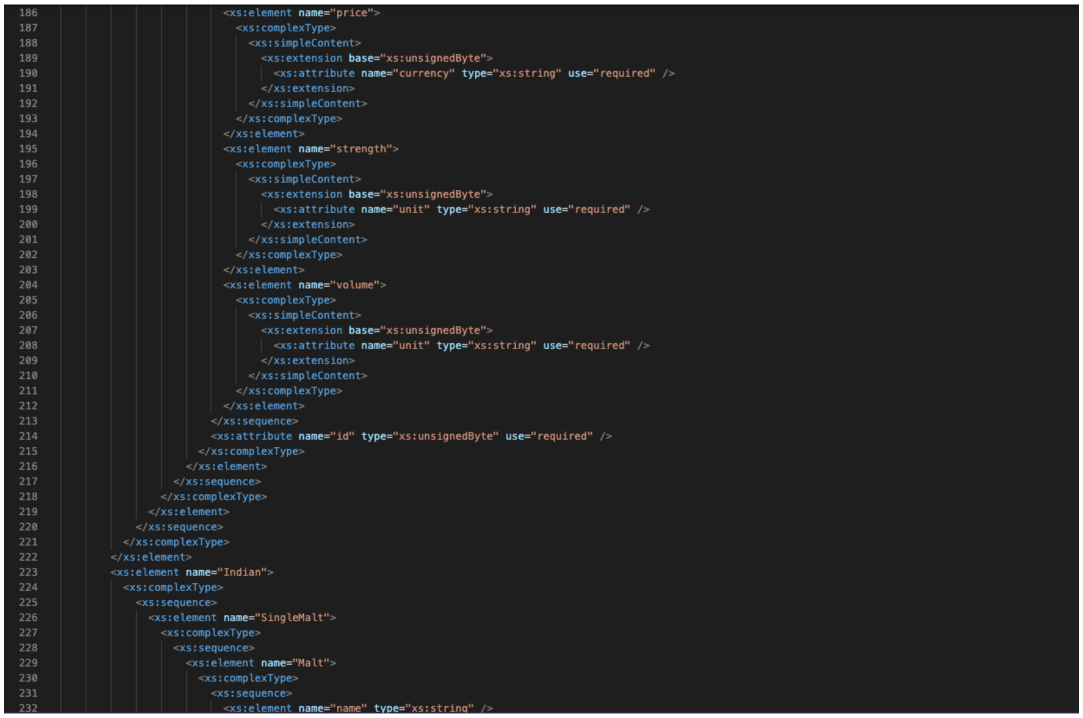
```
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="RyeWhiskey">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="Rye">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="name" type="xs:string" />
                      <xs:element name="price">
                        <xs:complexType>
                          <xs:simpleContent>
                            <xs:extension base="xs:decimal">
                              <xs:attribute name="currency" type="xs:string" use="required" />
                            </xs:extension>
                          </xs:simpleContent>
                        </xs:complexType>
                      </xs:element>
                      <xs:element name="strength">
                        <xs:complexType>
                          <xs:simpleContent>
                            <xs:extension base="xs:unsignedByte">
                              <xs:attribute name="unit" type="xs:string" use="required" />
                            </xs:extension>
                          </xs:simpleContent>
                        </xs:complexType>
                      </xs:element>
                      <xs:element name="volume">
                        <xs:complexType>
                          <xs:simpleContent>
                            <xs:extension base="xs:unsignedShort">
                              <xs:attribute name="unit" type="xs:string" use="required" />
                            </xs:extension>
                          </xs:simpleContent>
                        </xs:complexType>
                      </xs:element>
                    </xs:sequence>
                    <xs:attribute name="id" type="xs:unsignedByte" use="required" />
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
          <xs:element name="TennesseWhiskey">
            <xs:complexType>
              <xs:sequence>
```



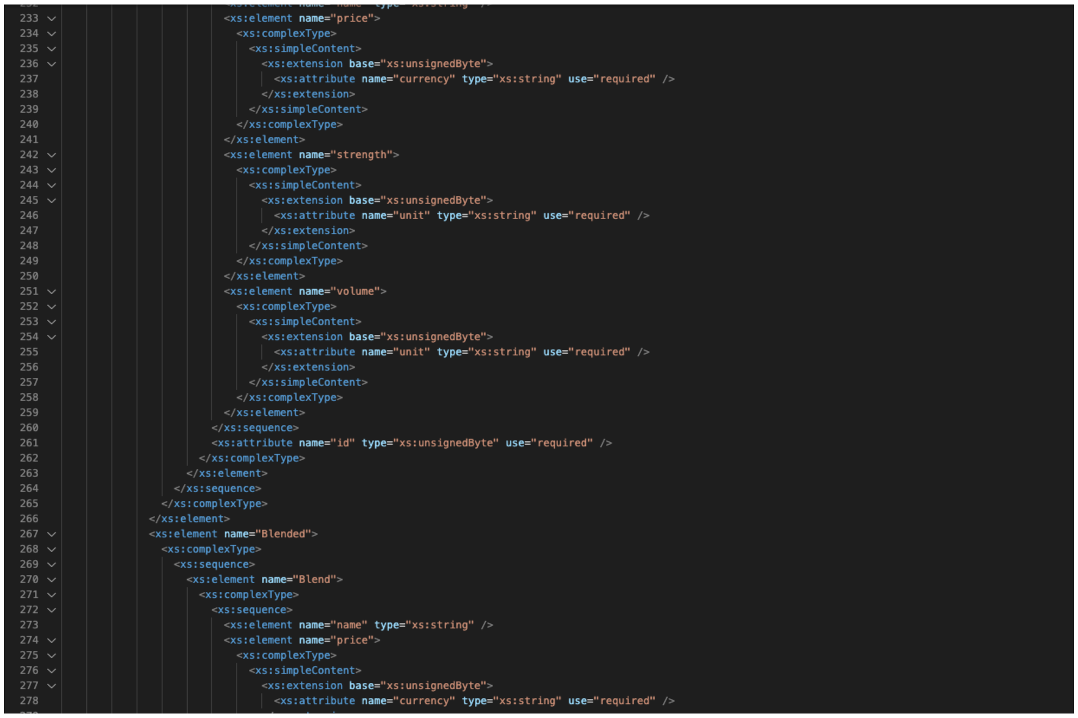
```
              <xs:sequence>
                <xs:element name="Tennesse">
                  <xs:complexType>
                    <xs:sequence>
                      <xs:element name="name" type="xs:string" />
                      <xs:element name="price">
                        <xs:complexType>
                          <xs:simpleContent>
                            <xs:extension base="xs:decimal">
                              <xs:attribute name="currency" type="xs:string" use="required" />
                            </xs:extension>
                          </xs:simpleContent>
                        </xs:complexType>
                      </xs:element>
                      <xs:element name="strength">
                        <xs:complexType>
                          <xs:simpleContent>
                            <xs:extension base="xs:unsignedByte">
                              <xs:attribute name="unit" type="xs:string" use="required" />
                            </xs:extension>
                          </xs:simpleContent>
                        </xs:complexType>
                      </xs:element>
                      <xs:element name="volume">
                        <xs:complexType>
                          <xs:simpleContent>
                            <xs:extension base="xs:unsignedShort">
                              <xs:attribute name="unit" type="xs:string" use="required" />
                            </xs:extension>
                          </xs:simpleContent>
                        </xs:complexType>
                      </xs:element>
                    </xs:sequence>
                    <xs:attribute name="id" type="xs:unsignedByte" use="required" />
                  </xs:complexType>
                </xs:element>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Irish">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="IrishWhiskey">
            <xs:complexType>
```



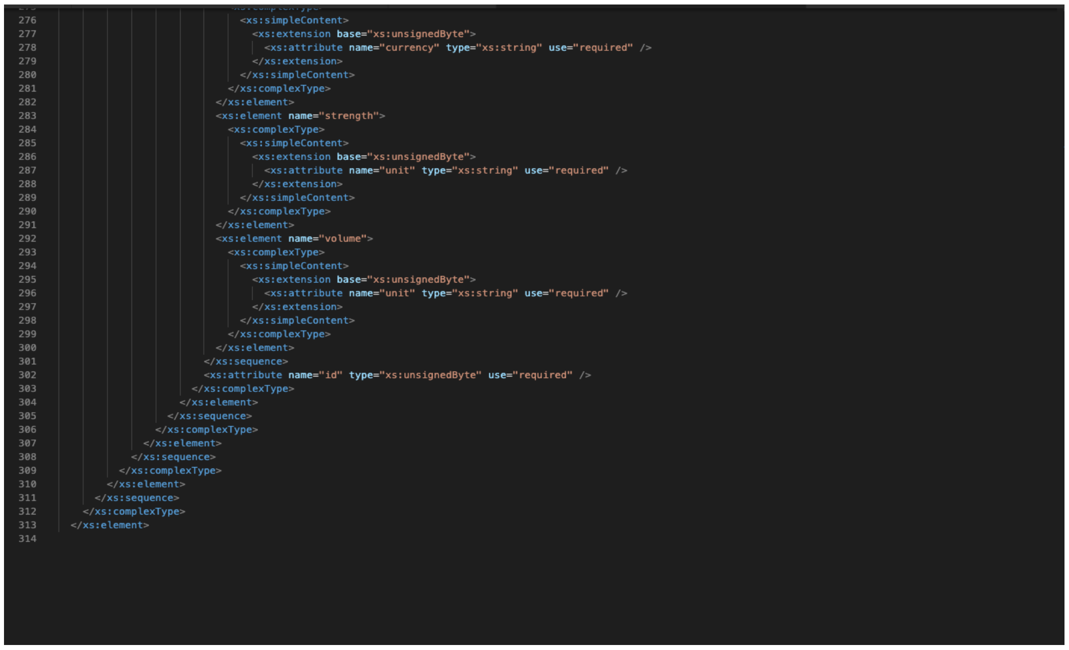
```
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Ire">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="name" type="xs:string" />
                          <xs:element name="price">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="currency" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="strength">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="volume">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:unsignedByte" use="required" />
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="Poitin">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Poi">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="name" type="xs:string" />
```



```
                          <xs:element name="price">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="currency" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="strength">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="volume">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:unsignedByte" use="required" />
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="Indian">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="SingleMalt">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Malt">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="name" type="xs:string" />
```

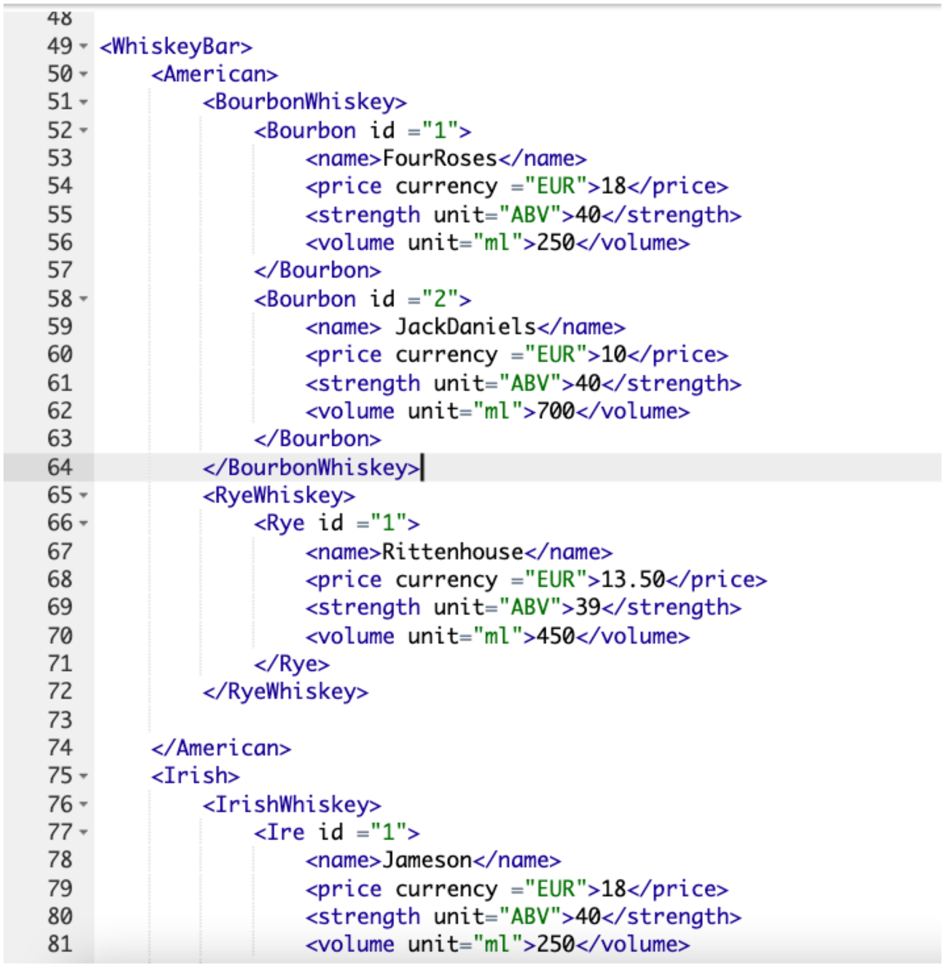


```
                          <xs:element name="price">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="currency" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="strength">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="volume">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:unsignedByte" use="required" />
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="Blended">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Blend">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="name" type="xs:string" />
                          <xs:element name="price">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="currency" type="xs:string" use="required" />
```

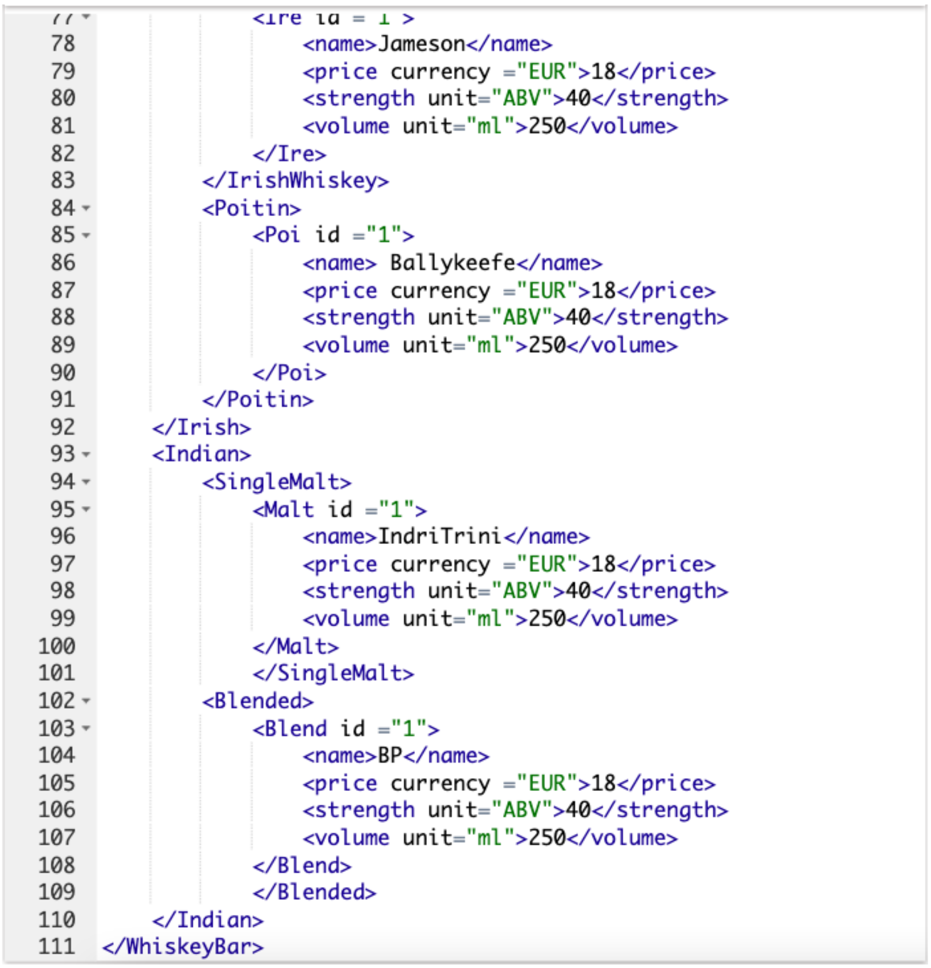


```
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="currency" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="strength">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="volume">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="unit" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:unsignedByte" use="required" />
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
```

3.The First XML document, in which each unbounded whiskey node has at least one object with name, price, strength, volume and its ID. Here in the first example, Tennesse whiskey wasn't added.

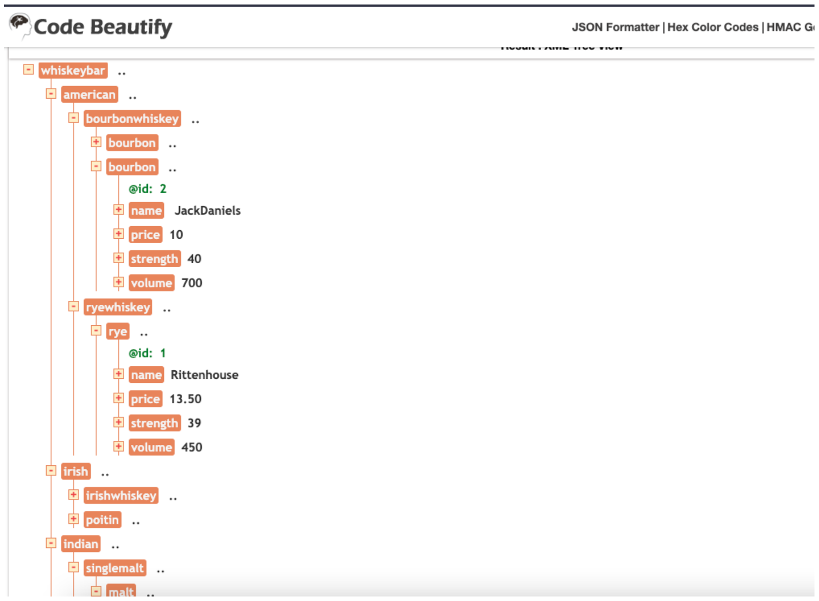


```xml
<WhiskeyBar>
    <American>
        <BourbonWhiskey>
            <Bourbon id ="1">
                <name>FourRoses</name>
                <price currency ="EUR">18</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">250</volume>
            </Bourbon>
            <Bourbon id ="2">
                <name> JackDaniels</name>
                <price currency ="EUR">10</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">700</volume>
            </Bourbon>
        </BourbonWhiskey>
        <RyeWhiskey>
            <Rye id ="1">
                <name>Rittenhouse</name>
                <price currency ="EUR">13.50</price>
                <strength unit="ABV">39</strength>
                <volume unit="ml">450</volume>
            </Rye>
        </RyeWhiskey>

    </American>
    <Irish>
        <IrishWhiskey>
            <Ire id ="1">
                <name>Jameson</name>
                <price currency ="EUR">18</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">250</volume>
```



```
            <Ire id ="1">
                <name>Jameson</name>
                <price currency ="EUR">18</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">250</volume>
            </Ire>
        </IrishWhiskey>
        <Poitin>
            <Poi id ="1">
                <name> Ballykeefe</name>
                <price currency ="EUR">18</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">250</volume>
            </Poi>
        </Poitin>
    </Irish>
    <Indian>
        <SingleMalt>
            <Malt id ="1">
                <name>IndriTrini</name>
                <price currency ="EUR">18</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">250</volume>
            </Malt>
            </SingleMalt>
        <Blended>
            <Blend id ="1">
                <name>BP</name>
                <price currency ="EUR">18</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">250</volume>
            </Blend>
            </Blended>
    </Indian>
</WhiskeyBar>
```

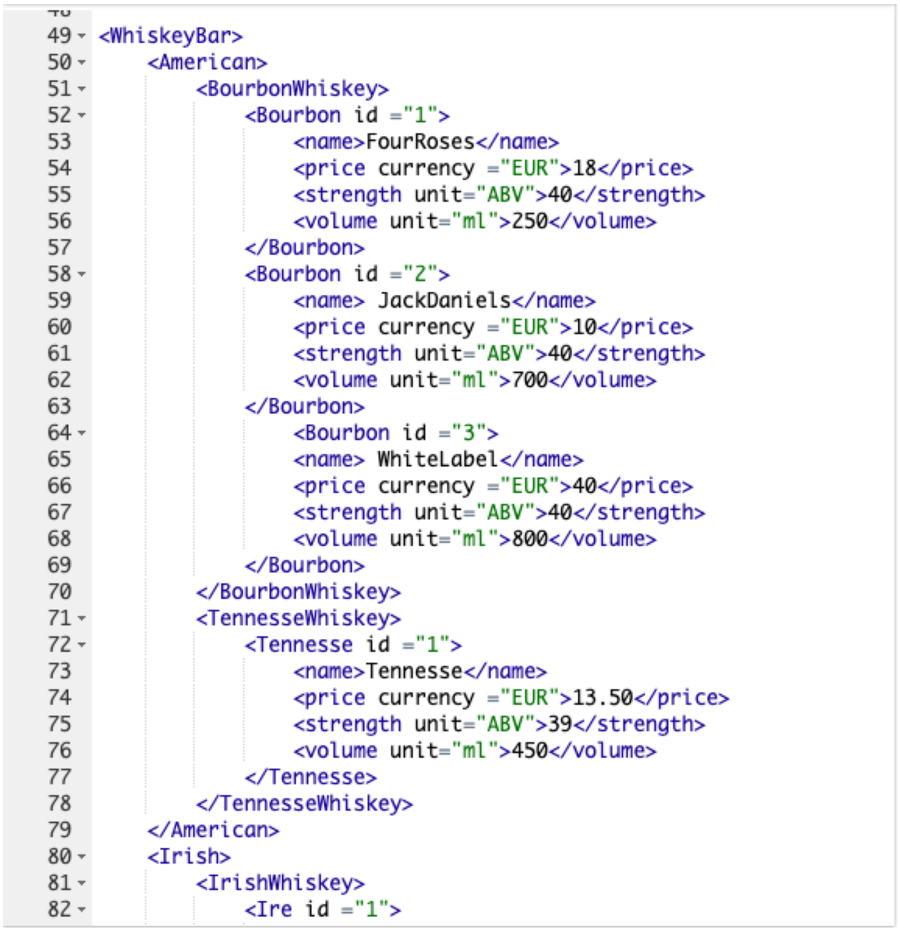
XML tree generated of the 1st XML example is below:


Code Beautify JSON Formatter | Hex Color Codes | HMAC G

```
whiskeybar ..
  american ..
    bourbonwhiskey ..
      bourbon ..
      bourbon ..
        @id: 2
        name JackDaniels
        price 10
        strength 40
        volume 700
    ryewhiskey ..
      rye ..
        @id: 1
        name Rittenhouse
        price 13.50
        strength 39
        volume 450
  irish ..
    irishwhiskey ..
    poitin ..
  indian ..
    singlemalt ..
      malt ..
```

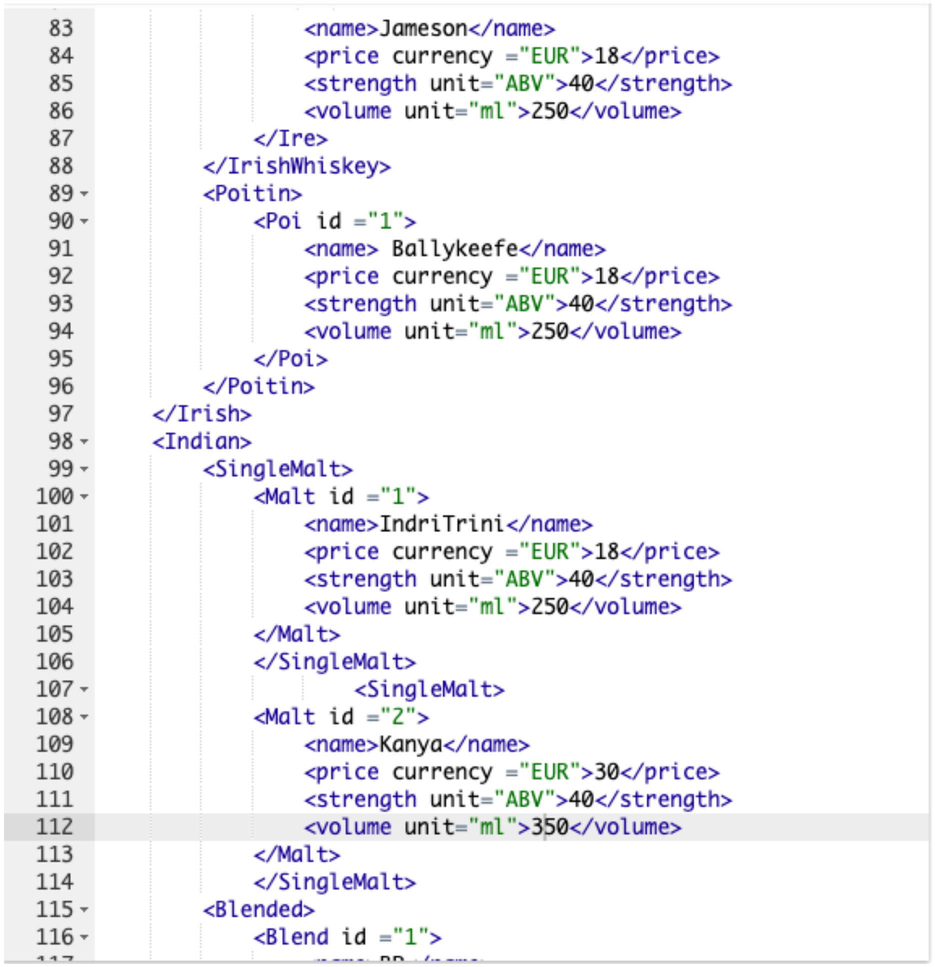
American whiskey in this frame is explored in this image.

XML example 2:



```
<WhiskeyBar>
    <American>
        <BourbonWhiskey>
            <Bourbon id ="1">
                <name>FourRoses</name>
                <price currency ="EUR">18</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">250</volume>
            </Bourbon>
            <Bourbon id ="2">
                <name> JackDaniels</name>
                <price currency ="EUR">10</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">700</volume>
            </Bourbon>
                <Bourbon id ="3">
                <name> WhiteLabel</name>
                <price currency ="EUR">40</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">800</volume>
            </Bourbon>
        </BourbonWhiskey>
        <TennesseWhiskey>
            <Tennesse id ="1">
                <name>Tennesse</name>
                <price currency ="EUR">13.50</price>
                <strength unit="ABV">39</strength>
                <volume unit="ml">450</volume>
            </Tennesse>
        </TennesseWhiskey>
    </American>
    <Irish>
        <IrishWhiskey>
            <Ire id ="1">
```

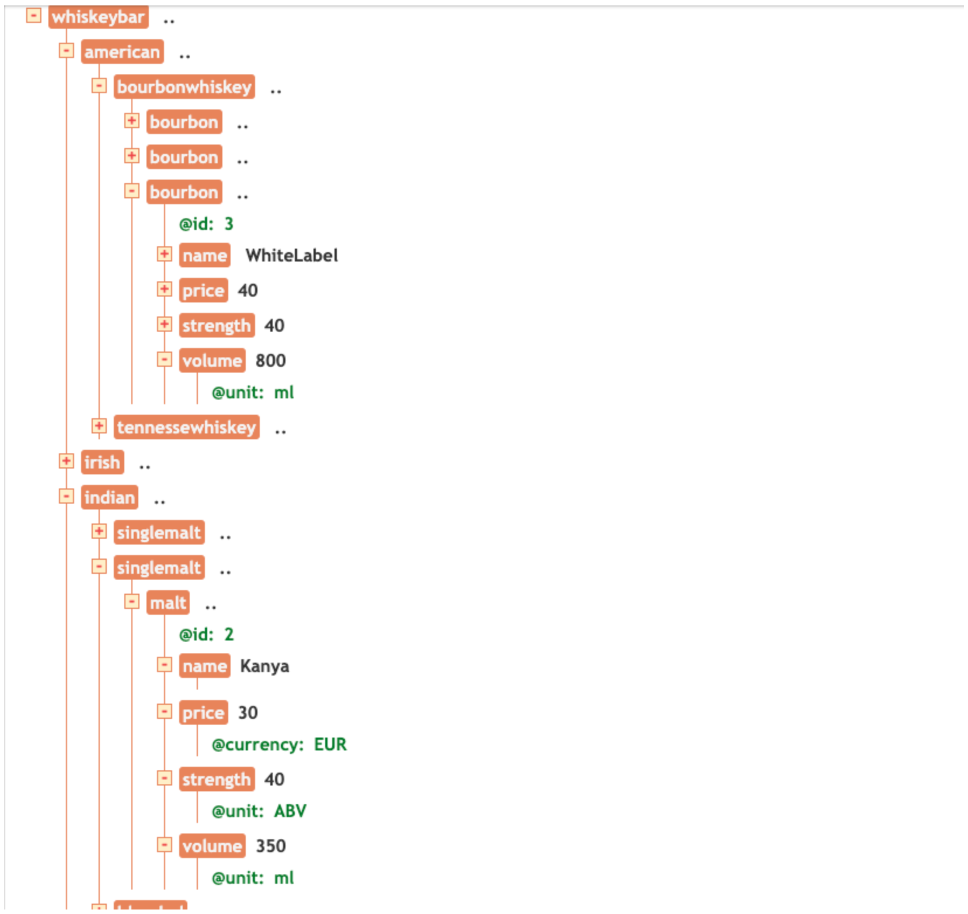
- Here on Line-64 a new bourbon ‘WhiteLabel’ is added, and Rye is absent.



```
 83                  <name>Jameson</name>
 84                  <price currency ="EUR">18</price>
 85                  <strength unit="ABV">40</strength>
 86                  <volume unit="ml">250</volume>
 87              </Ire>
 88          </IrishWhiskey>
 89          <Poitin>
 90              <Poi id ="1">
 91                  <name> Ballykeefe</name>
 92                  <price currency ="EUR">18</price>
 93                  <strength unit="ABV">40</strength>
 94                  <volume unit="ml">250</volume>
 95              </Poi>
 96          </Poitin>
 97      </Irish>
 98      <Indian>
 99          <SingleMalt>
100              <Malt id ="1">
101                  <name>IndriTrini</name>
102                  <price currency ="EUR">18</price>
103                  <strength unit="ABV">40</strength>
104                  <volume unit="ml">250</volume>
105              </Malt>
106              </SingleMalt>
107                      <SingleMalt>
108              <Malt id ="2">
109                  <name>Kanya</name>
110                  <price currency ="EUR">30</price>
111                  <strength unit="ABV">40</strength>
112                  <volume unit="ml">350</volume>
113              </Malt>
114              </SingleMalt>
115          <Blended>
116              <Blend id ="1">
```

- On line 108 the 2nd Indian single Malt ‘Kanya’ is added.

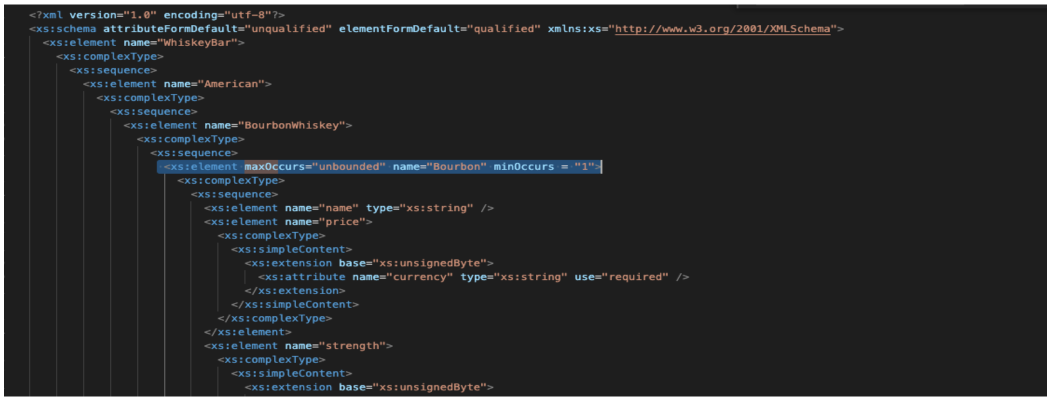
The new XML Tree view of 2nd example highlighting Bourbon and Single Malt is below:



```
- whiskeybar ..
  - american ..
    - bourbonwhiskey ..
      + bourbon ..
      + bourbon ..
      - bourbon ..
          @id: 3
        + name WhiteLabel
        + price 40
        + strength 40
        - volume 800
            @unit: ml
    + tennessewhiskey ..
  + irish ..
  - indian ..
    + singlemalt ..
    - singlemalt ..
      - malt ..
          @id: 2
        - name Kanya
        - price 30
            @currency: EUR
        - strength 40
            @unit: ABV
        - volume 350
            @unit: ml
```

Partial Examples of XSD catching,

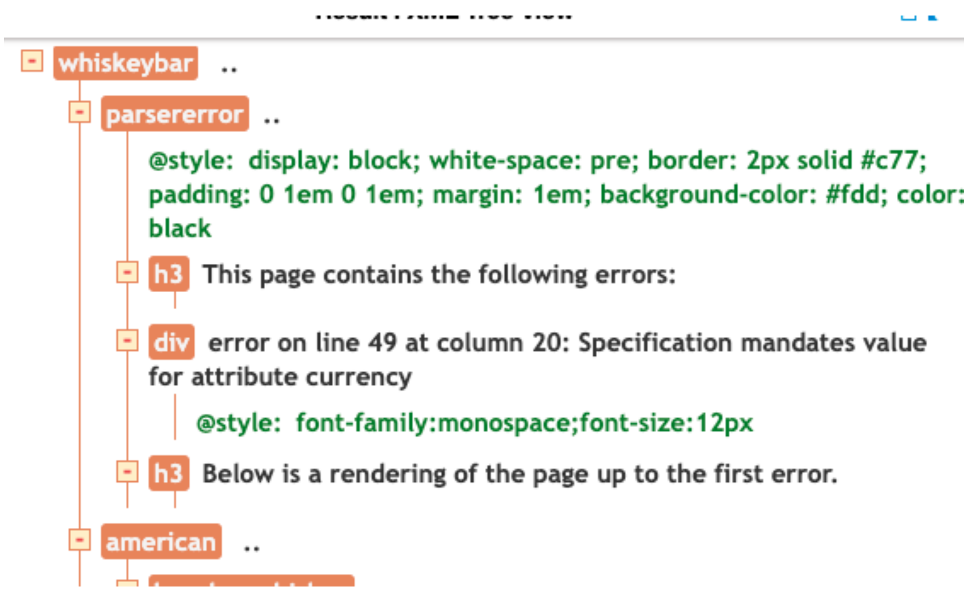
- In XSD in each occurrence, Minimum Occurrence(minOccurs) and Maximum Occurrence(maxOccurs) can be defined and will further prevent the XML from being updated or made unless the criterion is specified.



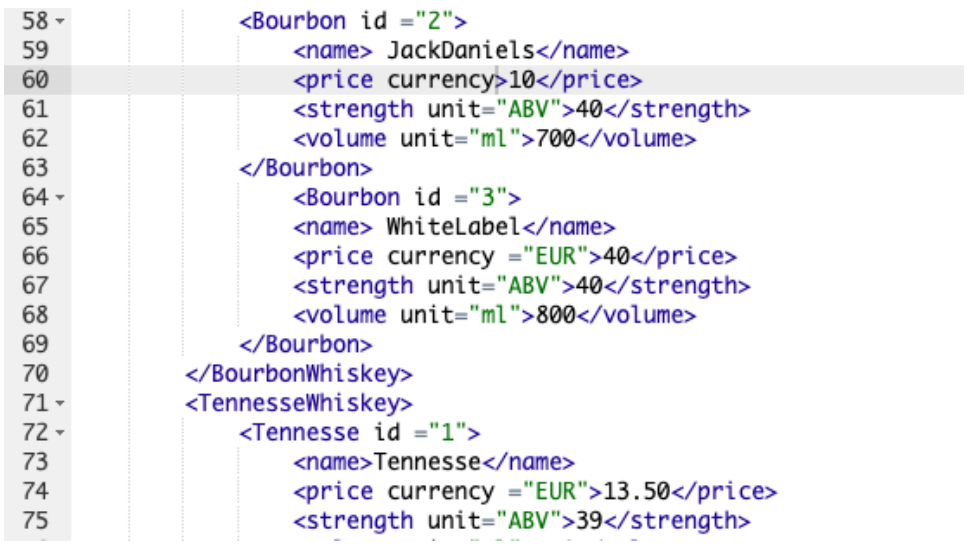
```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="WhiskeyBar">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="American">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="BourbonWhiskey">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element maxOccurs="unbounded" name="Bourbon" minOccurs = "1">
                      <xs:complexType>
                        <xs:sequence>
                          <xs:element name="name" type="xs:string" />
                          <xs:element name="price">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
                                  <xs:attribute name="currency" type="xs:string" use="required" />
                                </xs:extension>
                              </xs:simpleContent>
                            </xs:complexType>
                          </xs:element>
                          <xs:element name="strength">
                            <xs:complexType>
                              <xs:simpleContent>
                                <xs:extension base="xs:unsignedByte">
```

When minOccurs =”1” of bourbon is defined, adding more than one bourbon is unable to validate the XML.

- Not defining the Price – Un-identifying the tokenID or price again does not validate the XML properly.



```
whiskeybar ..
    parsererror ..
        @style:  display: block; white-space: pre; border: 2px solid #c77;
        padding: 0 1em 0 1em; margin: 1em; background-color: #fdd; color:
        black
        h3 This page contains the following errors:
        div error on line 49 at column 20: Specification mandates value
        for attribute currency
            @style:  font-family:monospace;font-size:12px
        h3 Below is a rendering of the page up to the first error.
    american ..
```



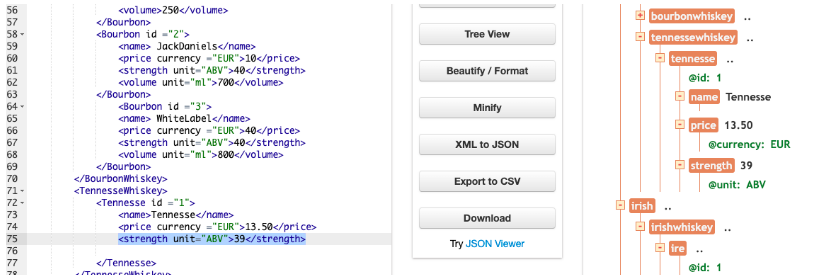
```
        <Bourbon id ="2">
            <name> JackDaniels</name>
            <price currency>10</price>
            <strength unit="ABV">40</strength>
            <volume unit="ml">700</volume>
        </Bourbon>
            <Bourbon id ="3">
            <name> WhiteLabel</name>
            <price currency ="EUR">40</price>
            <strength unit="ABV">40</strength>
            <volume unit="ml">800</volume>
        </Bourbon>
    </BourbonWhiskey>
    <TennesseWhiskey>
        <Tennesse id ="1">
            <name>Tennesse</name>
            <price currency ="EUR">13.50</price>
            <strength unit="ABV">39</strength>
```

3.Putting - !ATTLIST volume unit (ml) #IMPLIED>’

Here in case if volume is forgotten it won't cause an error, Implied means variable volume isn't necessary, and thus each product can be added by not adding the volume.

However, in this example case, if querying is required, use of volume should not be encouraged for searching. #Implied can be implemented depending on the problem statement at hand.

The tree generated successfully, in the absence of volume in Tennesse-whiskey's Volume.

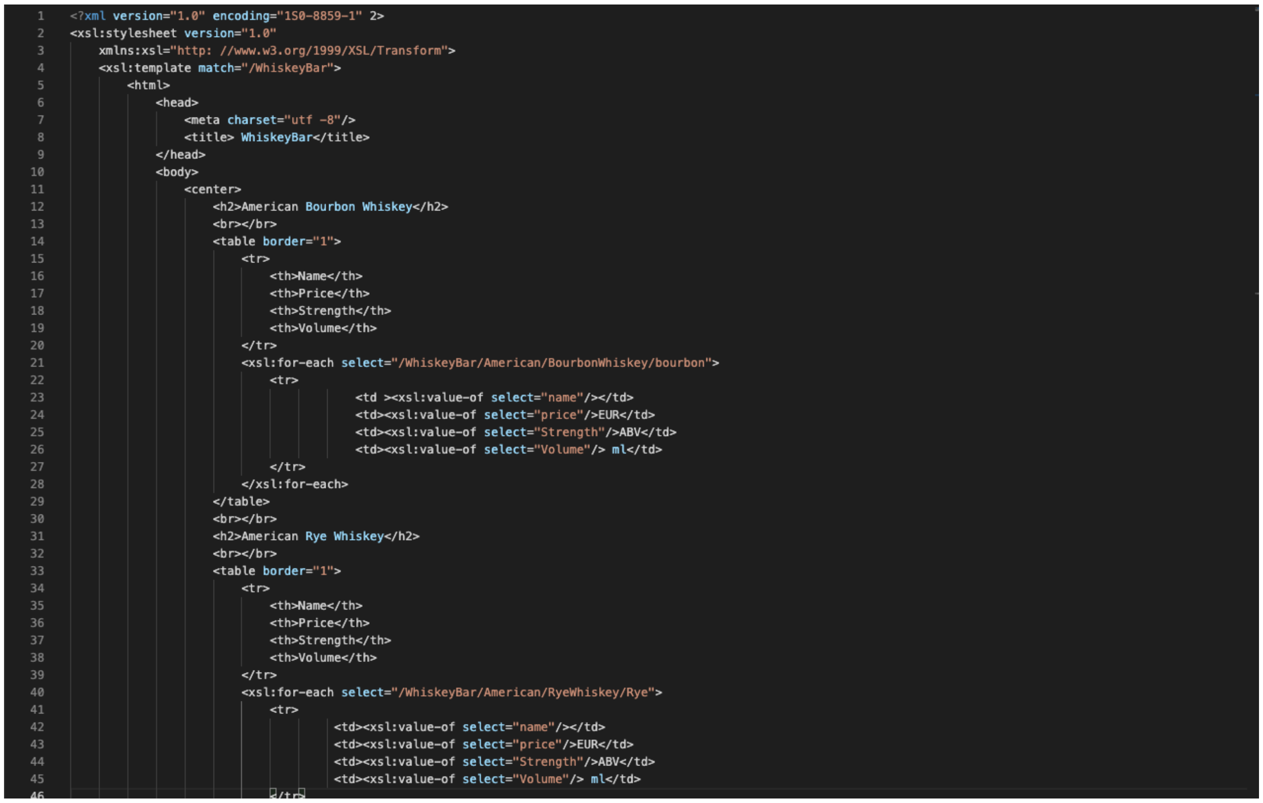


```
                <volume>250</volume>
            </Bourbon>
            <Bourbon id ="2">
                <name> JackDaniels</name>
                <price currency ="EUR">10</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">700</volume>
            </Bourbon>
                <Bourbon id ="3">
                <name> WhiteLabel</name>
                <price currency ="EUR">40</price>
                <strength unit="ABV">40</strength>
                <volume unit="ml">800</volume>
            </Bourbon>
        </BourbonWhiskey>
        <TennesseWhiskey>
            <Tennesse id ="1">
                <name>Tennesse</name>
                <price currency ="EUR">13.50</price>
                <strength unit="ABV">39</strength>

            </Tennesse>
```

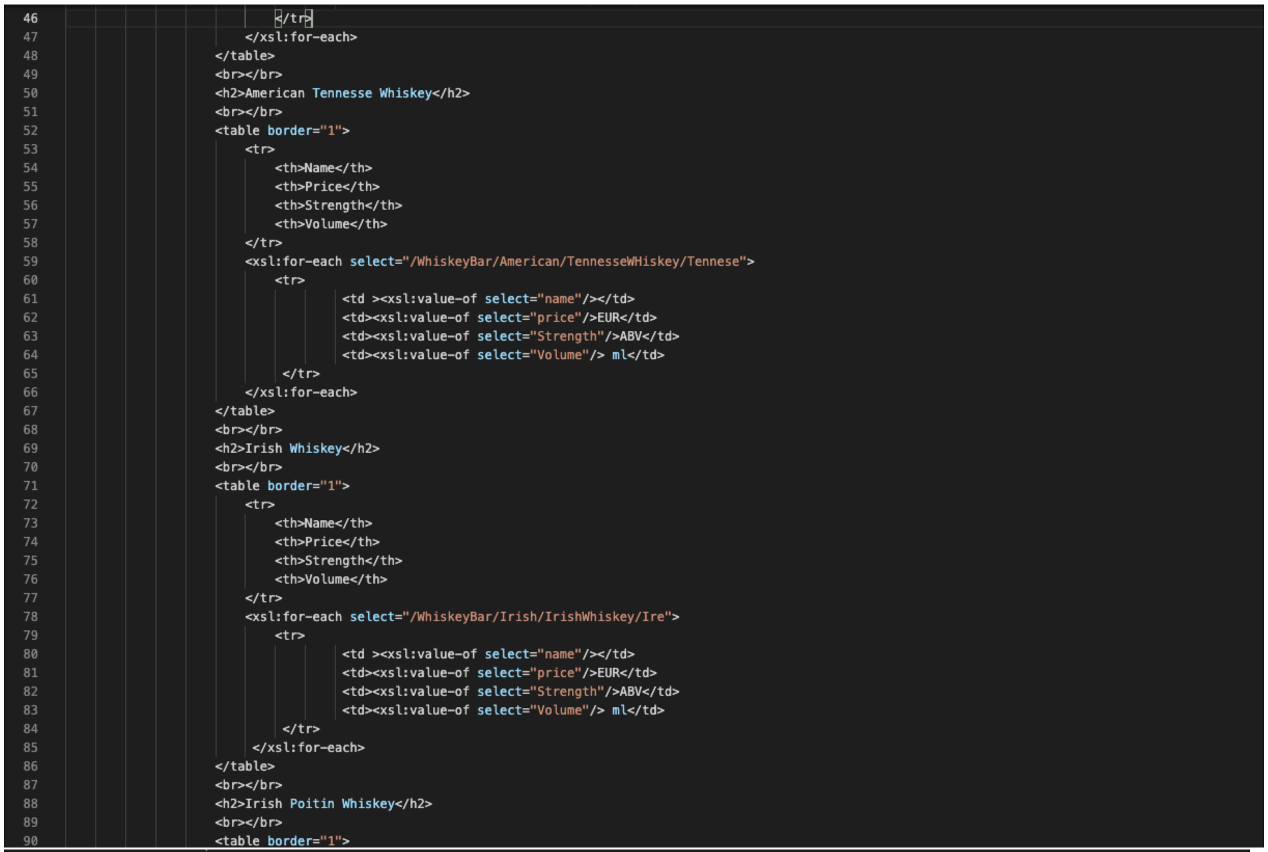
```
bourbonwhiskey ..
tennessewhiskey ..
  tennesse ..
    @id: 1
    name Tennesse
    price 13.50
      @currency: EUR
    strength 39
      @unit: ABV
irish ..
  irishwhiskey ..
    ire ..
      @id: 1
```

4.

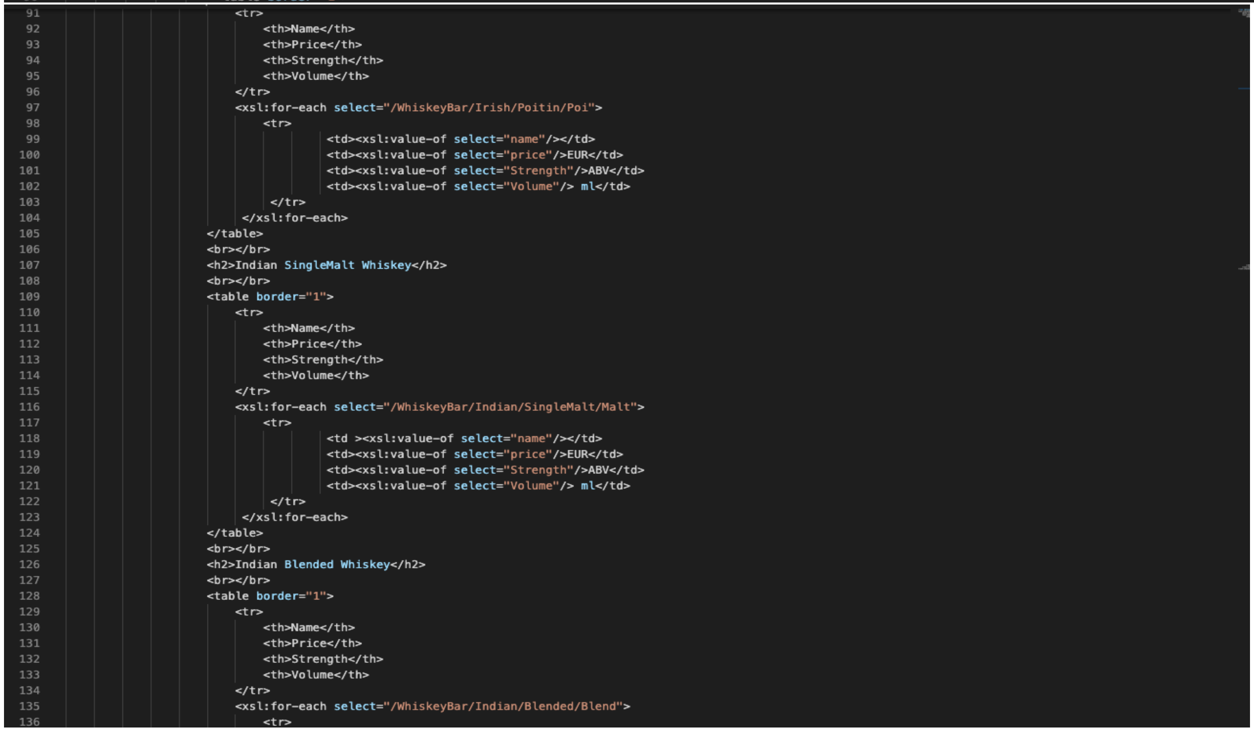
- `<xsl:template match="/WhiskeyBar">` defines the template of the page.
- Using similar tags in HTML to define the structure.
- Created the HTML tags first for of all the whiskeys.
- Using the tag -'<xsl:for-each>' in the .XSL file, and each tag individually links for the tables to the HTML file.
- After the .XSL file is tailored according to the .HTML file, to finally process the XML file.



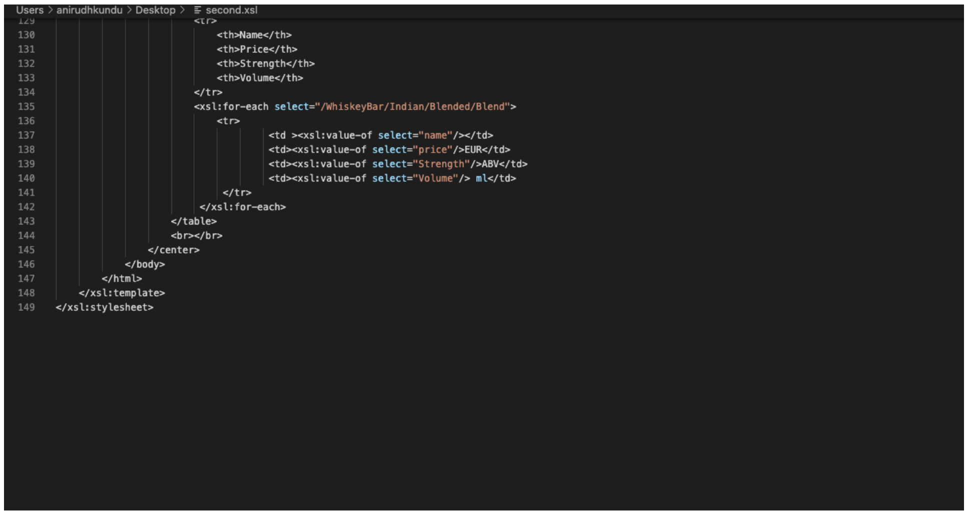
```
<?xml version="1.0" encoding="1S0-8859-1" 2>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http: //www.w3.org/1999/XSL/Transform">
    <xsl:template match="/WhiskeyBar">
        <html>
            <head>
                <meta charset="utf -8"/>
                <title> WhiskeyBar</title>
            </head>
            <body>
                <center>
                    <h2>American Bourbon Whiskey</h2>
                    <br></br>
                    <table border="1">
                        <tr>
                            <th>Name</th>
                            <th>Price</th>
                            <th>Strength</th>
                            <th>Volume</th>
                        </tr>
                        <xsl:for-each select="/WhiskeyBar/American/BourbonWhiskey/bourbon">
                            <tr>
                                        <td ><xsl:value-of select="name"/></td>
                                        <td><xsl:value-of select="price"/>EUR</td>
                                        <td><xsl:value-of select="Strength"/>ABV</td>
                                        <td><xsl:value-of select="Volume"/> ml</td>
                            </tr>
                        </xsl:for-each>
                    </table>
                    <br></br>
                    <h2>American Rye Whiskey</h2>
                    <br></br>
                    <table border="1">
                        <tr>
                            <th>Name</th>
                            <th>Price</th>
                            <th>Strength</th>
                            <th>Volume</th>
                        </tr>
                        <xsl:for-each select="/WhiskeyBar/American/RyeWhiskey/Rye">
                            <tr>
                                    <td><xsl:value-of select="name"/></td>
                                    <td><xsl:value-of select="price"/>EUR</td>
                                    <td><xsl:value-of select="Strength"/>ABV</td>
                                    <td><xsl:value-of select="Volume"/> ml</td>
                            </tr>
```



```
        </tr>
    </xsl:for-each>
</table>
<br></br>
<h2>American Tennesse Whiskey</h2>
<br></br>
<table border="1">
    <tr>
        <th>Name</th>
        <th>Price</th>
        <th>Strength</th>
        <th>Volume</th>
    </tr>
    <xsl:for-each select="/WhiskeyBar/American/TennesseWHiskey/Tennese">
        <tr>
                 <td ><xsl:value-of select="name"/></td>
                 <td><xsl:value-of select="price"/>EUR</td>
                 <td><xsl:value-of select="Strength"/>ABV</td>
                 <td><xsl:value-of select="Volume"/> ml</td>
         </tr>
    </xsl:for-each>
</table>
<br></br>
<h2>Irish Whiskey</h2>
<br></br>
<table border="1">
    <tr>
        <th>Name</th>
        <th>Price</th>
        <th>Strength</th>
        <th>Volume</th>
    </tr>
    <xsl:for-each select="/WhiskeyBar/Irish/IrishWhiskey/Ire">
        <tr>
                 <td ><xsl:value-of select="name"/></td>
                 <td><xsl:value-of select="price"/>EUR</td>
                 <td><xsl:value-of select="Strength"/>ABV</td>
                 <td><xsl:value-of select="Volume"/> ml</td>
         </tr>
     </xsl:for-each>
</table>
<br></br>
<h2>Irish Poitin Whiskey</h2>
<br></br>
<table border="1">
```

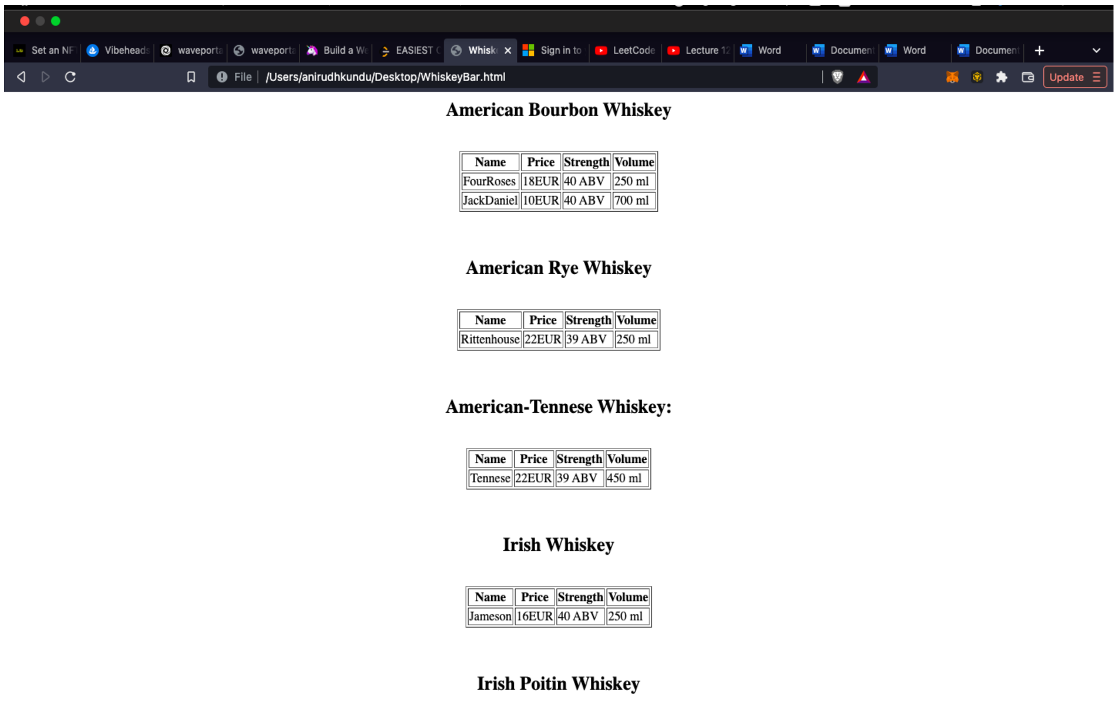


```
    <tr>
        <th>Name</th>
        <th>Price</th>
        <th>Strength</th>
        <th>Volume</th>
    </tr>
    <xsl:for-each select="/WhiskeyBar/Irish/Poitin/Poi">
        <tr>
                 <td><xsl:value-of select="name"/></td>
                 <td><xsl:value-of select="price"/>EUR</td>
                 <td><xsl:value-of select="Strength"/>ABV</td>
                 <td><xsl:value-of select="Volume"/> ml</td>
         </tr>
     </xsl:for-each>
</table>
<br></br>
<h2>Indian SingleMalt Whiskey</h2>
<br></br>
<table border="1">
    <tr>
        <th>Name</th>
        <th>Price</th>
        <th>Strength</th>
        <th>Volume</th>
    </tr>
    <xsl:for-each select="/WhiskeyBar/Indian/SingleMalt/Malt">
        <tr>
                 <td ><xsl:value-of select="name"/></td>
                 <td><xsl:value-of select="price"/>EUR</td>
                 <td><xsl:value-of select="Strength"/>ABV</td>
                 <td><xsl:value-of select="Volume"/> ml</td>
         </tr>
     </xsl:for-each>
</table>
<br></br>
<h2>Indian Blended Whiskey</h2>
<br></br>
<table border="1">
    <tr>
        <th>Name</th>
        <th>Price</th>
        <th>Strength</th>
        <th>Volume</th>
    </tr>
    <xsl:for-each select="/WhiskeyBar/Indian/Blended/Blend">
        <tr>
```



```
                <tr>
                    <th>Name</th>
                    <th>Price</th>
                    <th>Strength</th>
                    <th>Volume</th>
                </tr>
                <xsl:for-each select="/WhiskeyBar/Indian/Blended/Blend">
                    <tr>
                             <td ><xsl:value-of select="name"/></td>
                             <td><xsl:value-of select="price"/>EUR</td>
                             <td><xsl:value-of select="Strength"/>ABV</td>
                             <td><xsl:value-of select="Volume"/> ml</td>
                     </tr>
                 </xsl:for-each>
            </table>
            <br></br>
        </center>
    </body>
  </html>
 </xsl:template>
</xsl:stylesheet>
```

Generating the page,



American Bourbon Whiskey

Name	Price	Strength	Volume
FourRoses	18EUR	40 ABV	250 ml
JackDaniel	10EUR	40 ABV	700 ml

American Rye Whiskey

Name	Price	Strength	Volume
Rittenhouse	22EUR	39 ABV	250 ml

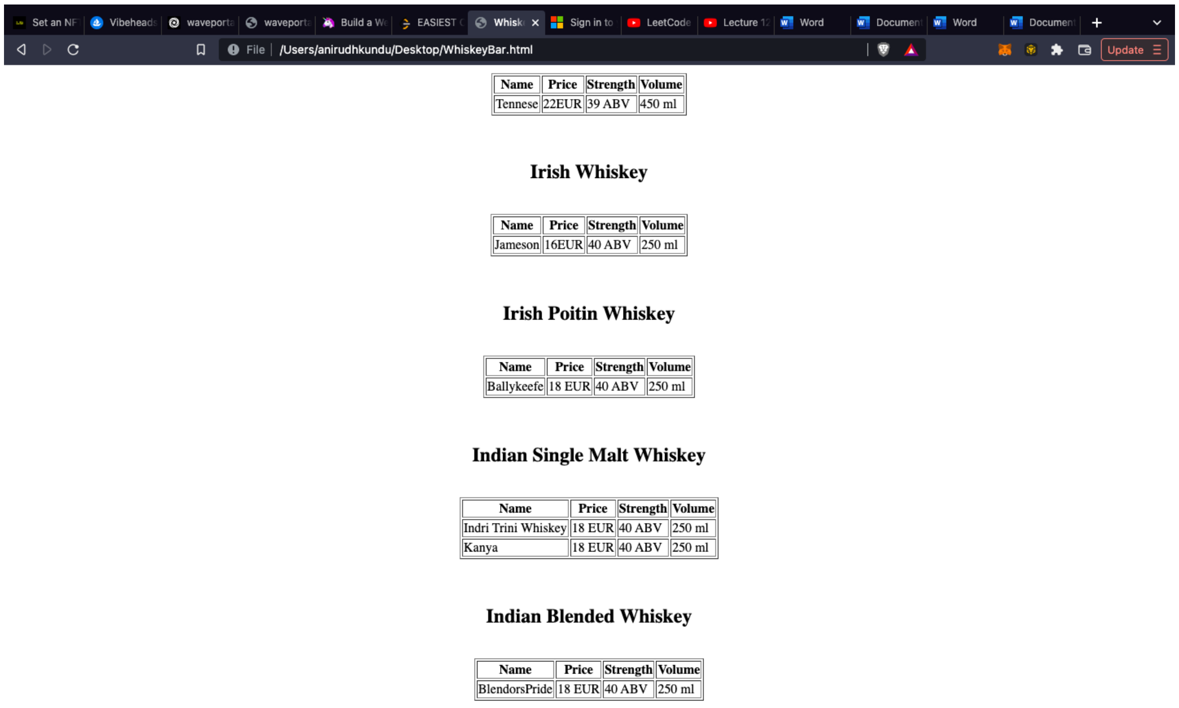
American-Tennese Whiskey:

Name	Price	Strength	Volume
Tennese	22EUR	39 ABV	450 ml

Irish Whiskey

Name	Price	Strength	Volume
Jameson	16EUR	40 ABV	250 ml

Irish Poitin Whiskey



Name	Price	Strength	Volume
Tennese	22EUR	39 ABV	450 ml

Irish Whiskey

Name	Price	Strength	Volume
Jameson	16EUR	40 ABV	250 ml

Irish Poitin Whiskey

Name	Price	Strength	Volume
Ballykeefe	18 EUR	40 ABV	250 ml

Indian Single Malt Whiskey

Name	Price	Strength	Volume
Indri Trini Whiskey	18 EUR	40 ABV	250 ml
Kanya	18 EUR	40 ABV	250 ml

Indian Blended Whiskey

Name	Price	Strength	Volume
BlendorsPride	18 EUR	40 ABV	250 ml

2. The first idea of a World Wide Web was designed by Tim Berner Lee when he discovered the hypertext functionality and its ability to link multiple user databases while working at CERN. Tim Berner's Lee gifted plenty of visions and dreams to computer science, including a semantic web that was quintessentially a Universal Grammar for computers that the machines could use to communicate among themselves. Even though the linguistic concept of Universal Grammar by Noam Chomsky was proved eventually incompatible and incomplete later. The concept of a semantic web, however, can be a paradigm in querying and retrieving knowledge and information.

The World Wide Web has evolved massively, from its Origins at CERN in 1990 to its first going public and developing standards and the dot-com bubble crash and aftermath – to being of Web as a platform, the web2.0, the introduction of ethics and security, to then finally being at the present state of being defined as a system with the documents connected to other documents by hyperlinks or URIs, for the user to then search for knowledge or information by traversing through those documents on the internet.

Google and Bing search engines use multiple systems and technology for returning a query from a user. These search engines first use the strings as inputs from the user then use those characters and words to compare with a wide range of documents that are already indexed on the web.

While using the strings Google and Bing primarily use NLP and machine learning to try understanding some context of the statement input by the user. For returning the query, the search engine also analyzes some local, geopolitical, geographical, and some of the user's data which is primarily used for indexing the priority of search results according to the user. Even though google and Bing search engines both slightly different in their approach, operate responding to a query in a similar way of understanding the syntax of the statement(language) and operate on them via a set of rules and systems.

The process of applying Natural Language Processing on the statement is one of the methods currently employed to understand a certain degree of the semantics of the query. NLP uses the principles of machine learning I.e., predicting the syntax(mainly) and semantics of the query by using a neural network, which uses data to learn patterns in data and output functions accordingly. Google and Bing's data for language processing, is trained majorly in syntactic methods in which it traverses and labels data and predicts the next set of characters or words. However, the most syntactic and some semantic data of NLP used by these search engines use data that are labelled based on human interaction and thought.

Thus, a normal Google or Bing search engine query would return results mainly based on its syntax and where most similar users in the past have ended up being satisfied. At present they are one of the most sophisticated search engines, but these search engines lack certainty and clear logic when answering or discussing a query. Ideally, if an object or a concept is to be searched, the result should be a certain idea, that is built on an epistemological logic that is unequivocal. Google and Bing engines make the precision low, and a vocabulary sensitive web search.

This leads us to discuss the efficacy of a more semantic-based approach of information retrieval, and the semantic web.

The Semantic web focuses on building structured knowledge, that is scalable. It also understands objects and links them based on different systems, so it becomes less complex and easy to scale with the passage of time.

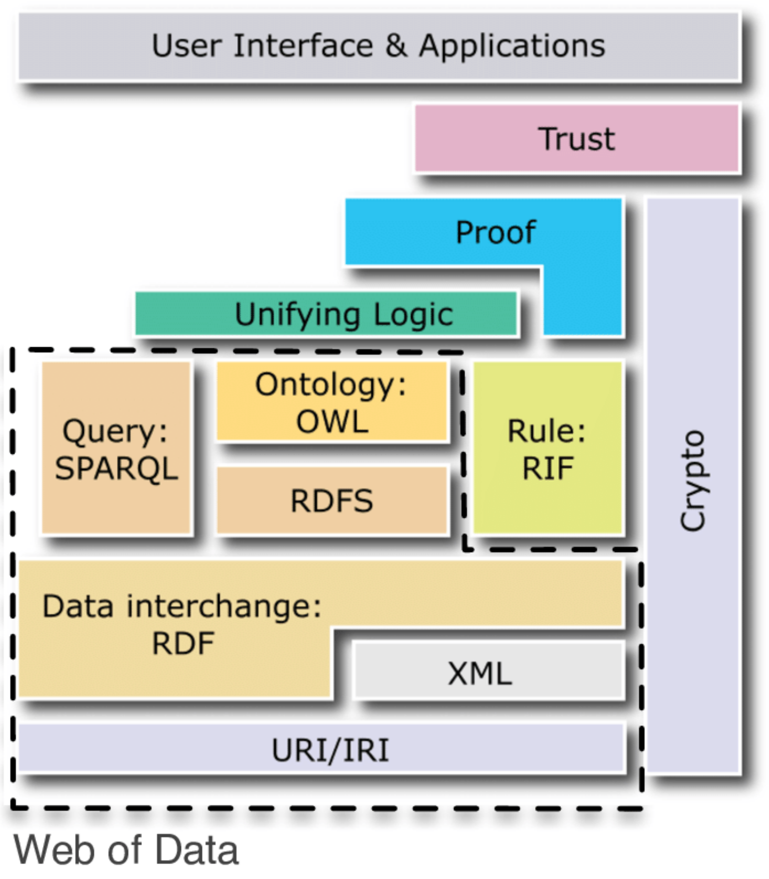
All the semantic web is ultimately built on the concept of RDFs that also acts as the building block forming the information exchange layer. RDF (Resource Description Framework) is a semantic triple, that consists of three entities subject --->Predicate--->object. RDFs are implemented in XML and were first conceptualized in 1996 by the W3C. The RDFs use URIs (similar to HTML-based web) to describe subjects and predicates but used IRIs for the object. IRIs are machine readable strings that link human written objects and entities on the web, which gives the expression of objects and entities being machine readable.

RDF triples now can be connected to other triples and form ontological graphs where the rules are defined by OWL. OWL (Web Ontology Language) enables now to form hierarchies and relationships of multiple objects in structured patterns and schemas, it again uses URIs to connect different ontologies and graphs.

If all the available data is linked together in a graph while simultaneously preserving the ability to query, it can be called linked data and is the ideal structure for the semantic web for publishing and consuming information. Presently search engines, data-based organizations and companies are promoting the semantic and structured method of storing and sharing knowledge.

Semantic-based approaches enable simplifying and scaling structured data, where the programming effort and complexity for extracting knowledge are least, making it an attractive idea to take interest in. Currently, semantic-based approaches are highly studied and researched from the various fields and directions that focus on understanding and building methods and approaches to better the semantic web services, where the processes are still based on some semi-formal natural language descriptions.

Semantic web has several contrasting limitations at present. Now we must see the limitations of our current understanding of the implementation of the semantic web.



The semantic web's stack architecture shows different layers of abstraction required to make a fully-fledged semantic. At present XML for URI's and Unicode for namespace are fully standardized, the RDF schema, ontologies have been mostly standardized, together with the existing RDF frameworks and Ontologies linked data can be generated and be used for querying and some information retrieval BBC's graph which uses BBC organisation's data, and dbpedia which in a way structures Wikipedia data to make information retrieval accurate and simple. However, to be an unequivocal information retrieval system as a whole, a set and clear logic standard that unifies the ontologies is not achieved as of now. Plenty of research and ideas are ongoing to understand a common unifying logic, or logics that are applicable in certain systems which are stable and universally true. However, eventually a system of proof and trust is to be designed and programmed such that a user can access knowledge on the semantic system to be then satisfied and have an answer which has an underlying logic and sufficient proof.

One other technical challenge is to scale and maintain the semantic web system efficiently and effectively. The power and ability to modify information on the graphs is more complex than the traditional current world wide web, as any information changed further changes the structure on the graph, this would be possible by cryptography which has been by far the most efficient technology for encryption and distribution of access, ability to create Dapps on the blockchain have now made it more realistic and approachable. Once a system is created and defined scaling would be less programming intensive and approach towards its total automation.

One other difficulty with the semantic web system is, at present ‘data’ is widely used and processed to make and generate relevant conclusive information. Thus, to make a fully logical semantic based search engine, some level of data-processing and analysis is required. Which points to the fact that semantic web might work along with our traditional approach to understanding data, both in a concoction to achieve a degree of trust and logic. The last challenge I believe is designing a UI interactable with the user, in which access to rights are distributed, while maintaining the web as free for all and up to standards of then ethical systems and values universally or locally achieved or accepted.

Apart from technical limitations, growth and understanding of linguistics and language processing, or even cognitive neuroscience of humans may define, update, or change the rules and design of the semantic web. Understanding language(linguistics) universally is still a challenge, and a lot to learn, understand, and discover about, which would play a key role in the long-term of how humans interact with semantic based information retrieval systems. At that point Tim Berner Lee’s idea of universal grammar would be achieved, and hopefully would pave the way for better ideas and designs to look forward to in the future.

However, in the next 10 years, unless there is a groundbreaking understanding in linguistics or a similar school, our current systems of the semantic web which builds on RDFs will continue and thrive.

Semantic web technologies will continue to give rise to its scalable technologies, that further enable the system to keep up with the changing environment in such a way that programming is minimum. Hence The Semantic based knowledge retrieval systems would be a stand-alone technology in the future, where mining new information would be more of a commodity and would not bottleneck the growth and innovation humanity has to provide. Semantic based systems would also convolve with other systems and technologies to aid to the benefit of the users, Intelligent Assistants is one of the applications of semantic based systems.

Quantum computers and processing may lead to larger scale access of the current semantic web services, where the limitation is computation and power for machine learning in NLP. Irrespective of innovative technologies that are to emerge, the fundamental idea of a scalable structured semantic information retrieval system would challenge the testament of time, until achieved.

References

[1] https://en.wikipedia.org/wiki/Semantic_Web

[2] Benjamins, R., Contreras, J., Corcho, Ó., & Gómez-Pérez, A. (2002). Six challenges for the Semantic Web. *KR 2002*. http://www.cs.man.ac.uk/~ocorcho/documents/KRR2002WS_BenjaminsEtAl.pdf

[3] https://apilama.com/2018/07/03/limitations-of-the-semantic-web/