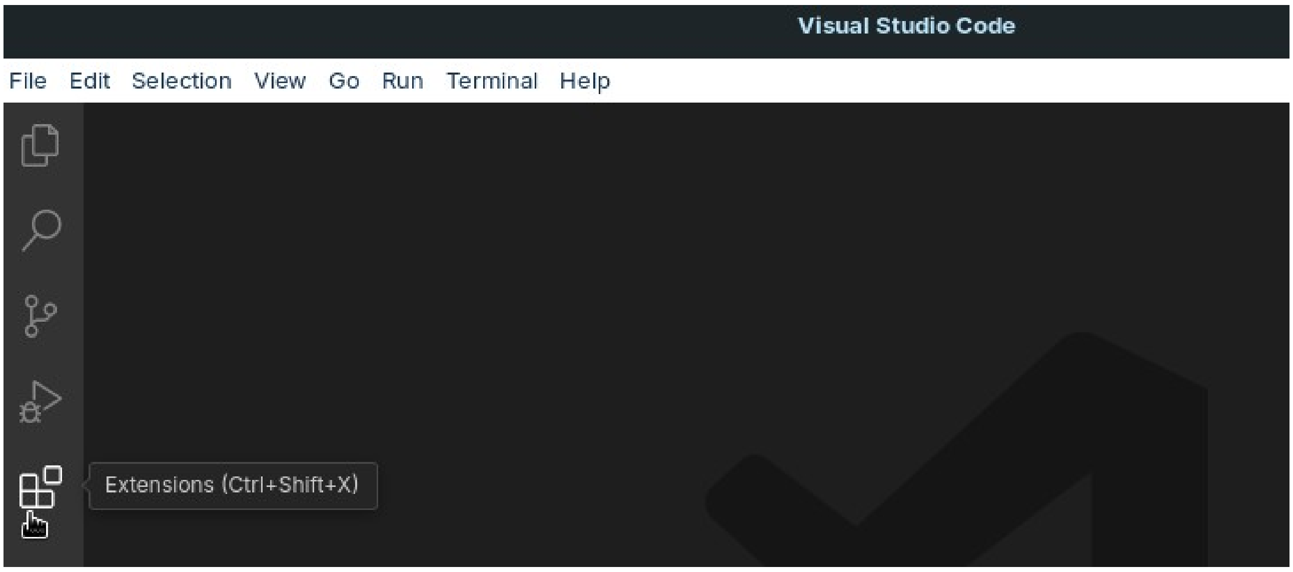
ISCG6420 Semester 1 2024

Exercise: XML Design Documents

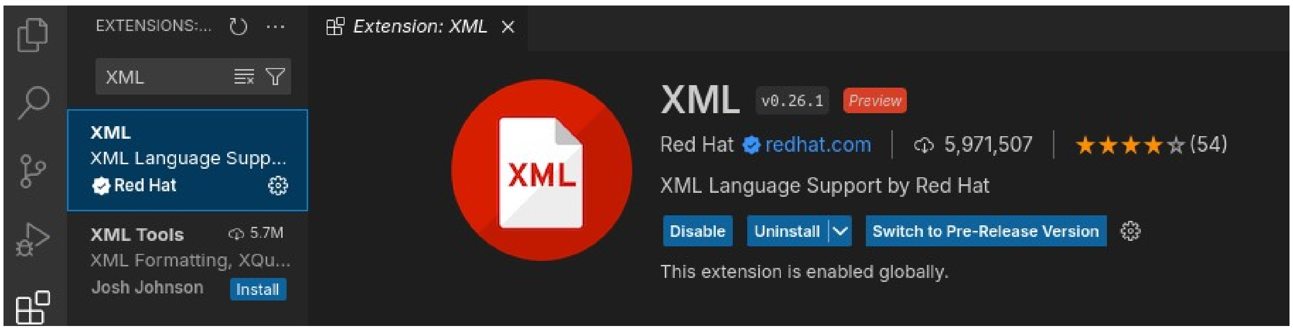
These instructions will take you through the process of creating XML design documents for Assessment 3 – Group Project.

Setup VSCode

Open VSCode and navigate to the Extensions menu.

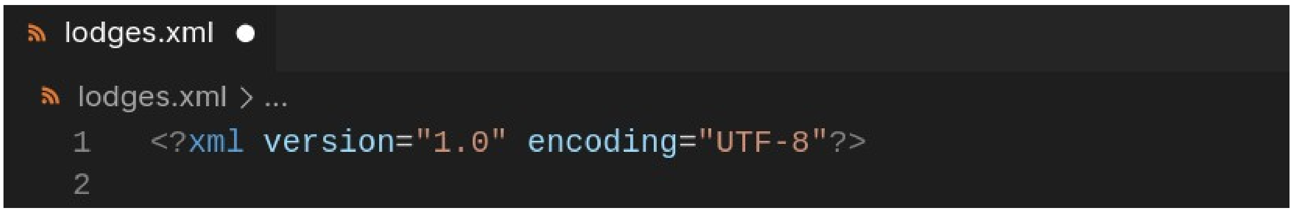


Use the search bar to search for "XML". Find the XML extension by RedHat and install it.



XML Document

Open a folder in VSCode and create a new file called **"lodges.xml"**. Inside the file declare the file type as XML version 1.0 with UTF-8 encoding:



lodges.xml ●

lodges.xml > ...

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

```

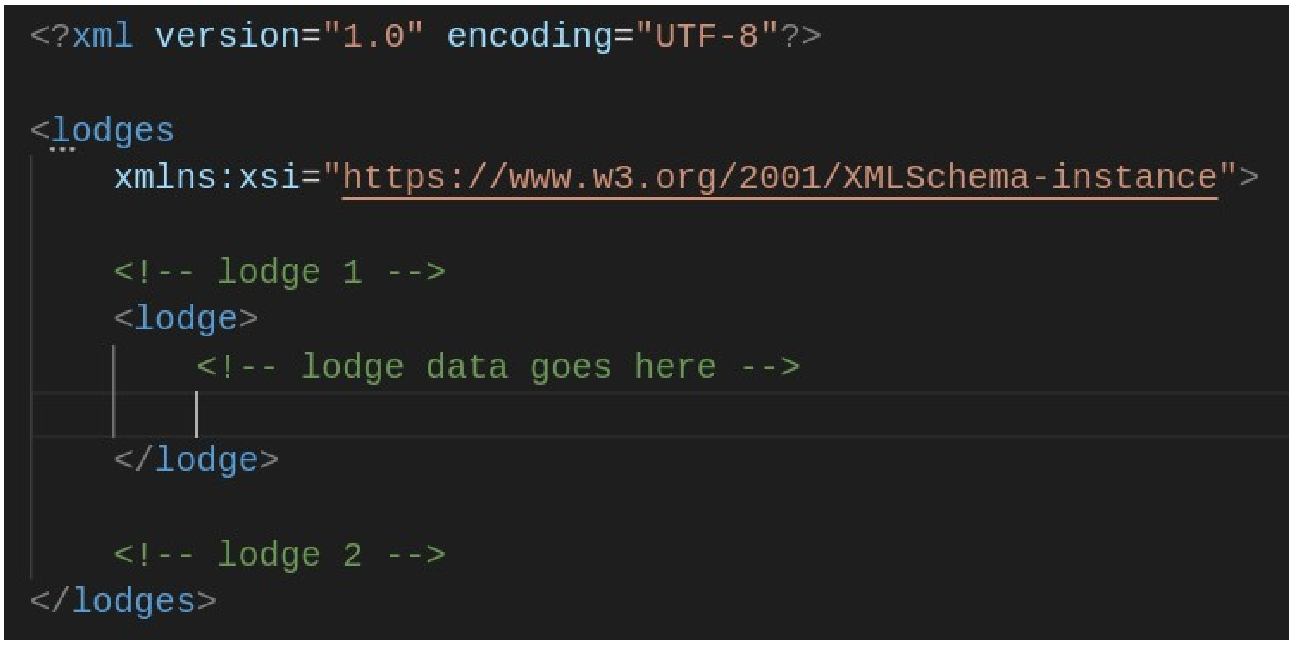
Using the data requirements in the Assessment 3 Group Project document (under Moodle -> Assessments), create XML elements to represent the data of the Piha Lodges. There are multiple lodges, each with multiple variables.

NOTE: It is recommended you discuss with your group partner before commiting to a data structure. It may help to brainstorm your structure with pen and paper, or a text editor. Consult your lecturer for feedback.

Below is a recommended starting structure for your XML document. The "lodges" element contains a list of lodges as child elements. Each lodge child will contain the data pertaining to that specific lodge. Each lodge should have the same set of variables but with differing values.

Use the W3 consortium's default schema namespace in your structure:

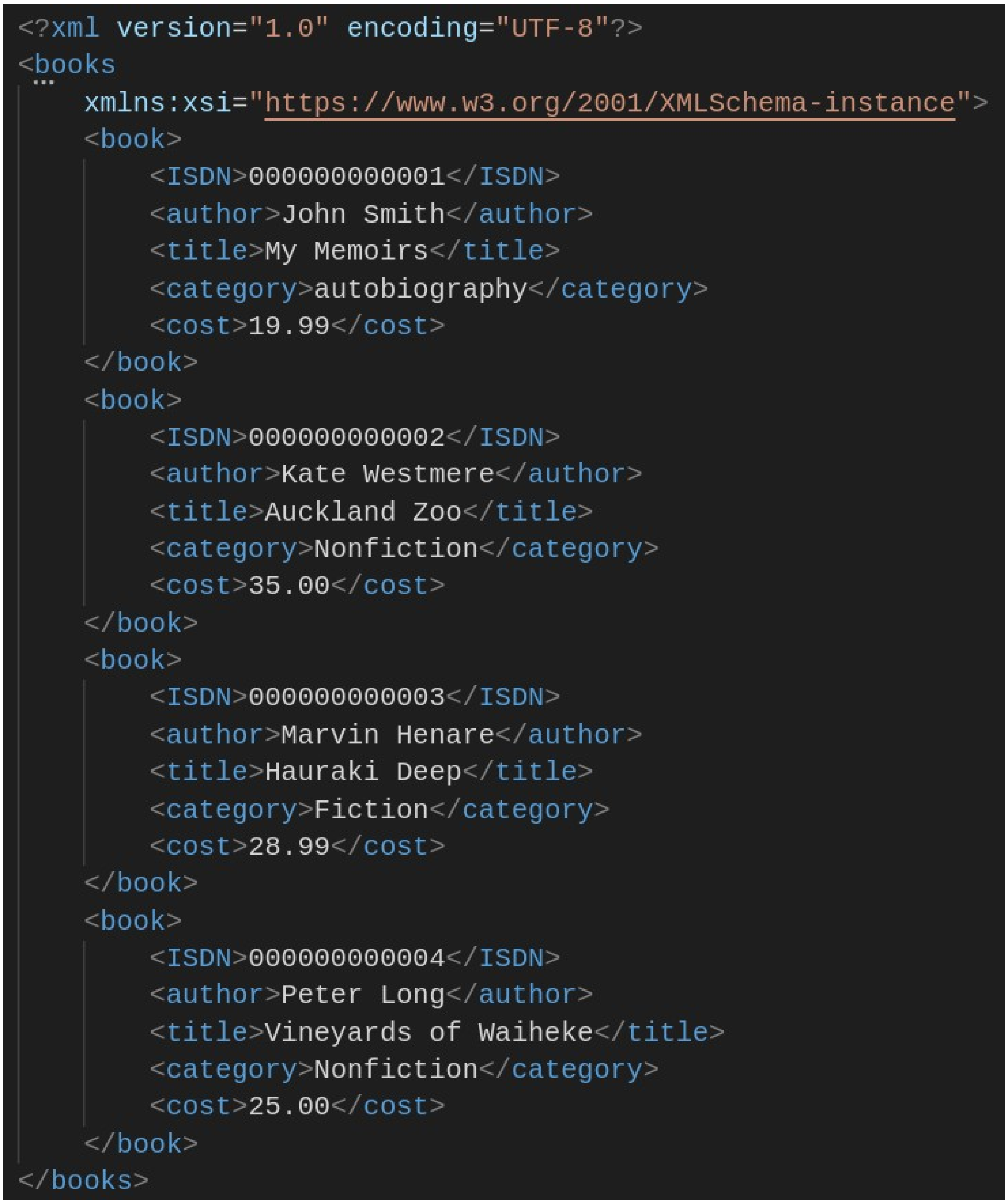
- https://www.w3.org/2001/XMLSchema-instance



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

<lodges
    xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance">

    <!-- lodge 1 -->
    <lodge>
        <!-- lodge data goes here -->

    </lodge>

    <!-- lodge 2 -->
</lodges>
```

XML Document Example

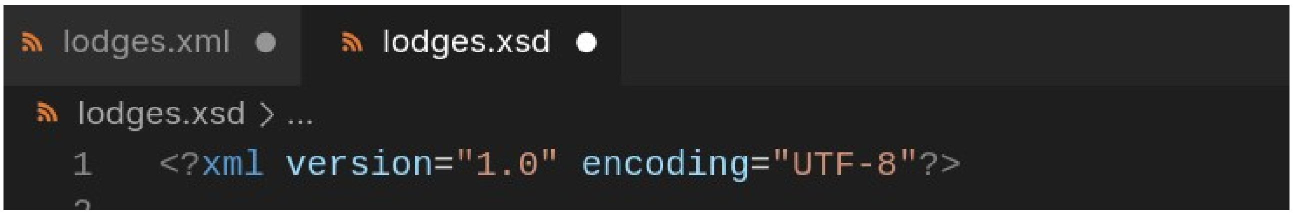
This is an example of a complete XML document for a collection of books. The topic of this document is entirely unrelated to Piha Lodge but shows the structural relationship of the books collection, the book element, and the book data elements.



```xml
<?xml version="1.0" encoding="UTF-8"?>
<books
    xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance">
    <book>
        <ISDN>000000000001</ISDN>
        <author>John Smith</author>
        <title>My Memoirs</title>
        <category>autobiography</category>
        <cost>19.99</cost>
    </book>
    <book>
        <ISDN>000000000002</ISDN>
        <author>Kate Westmere</author>
        <title>Auckland Zoo</title>
        <category>Nonfiction</category>
        <cost>35.00</cost>
    </book>
    <book>
        <ISDN>000000000003</ISDN>
        <author>Marvin Henare</author>
        <title>Hauraki Deep</title>
        <category>Fiction</category>
        <cost>28.99</cost>
    </book>
    <book>
        <ISDN>000000000004</ISDN>
        <author>Peter Long</author>
        <title>Vineyards of Waiheke</title>
        <category>Nonfiction</category>
        <cost>25.00</cost>
    </book>
</books>
```

XML Schema

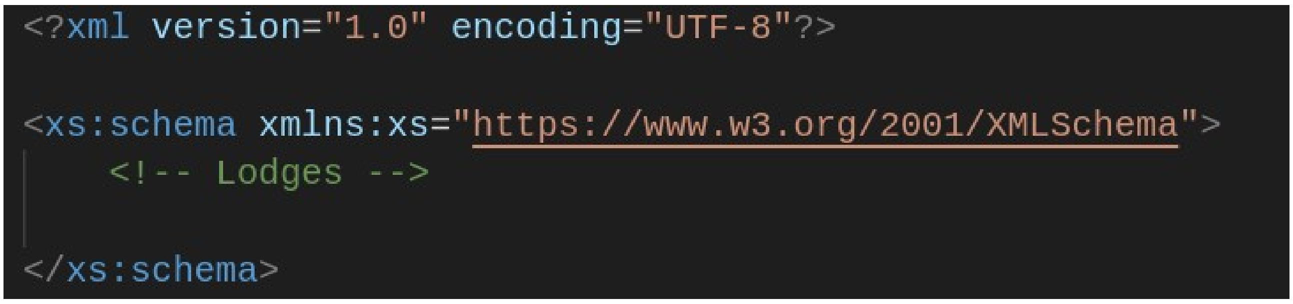
Create a second file called “**lodges.xsd**”. Inside the file declare the file type as **XML** version **1.0** with **UTF-8** encoding.



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

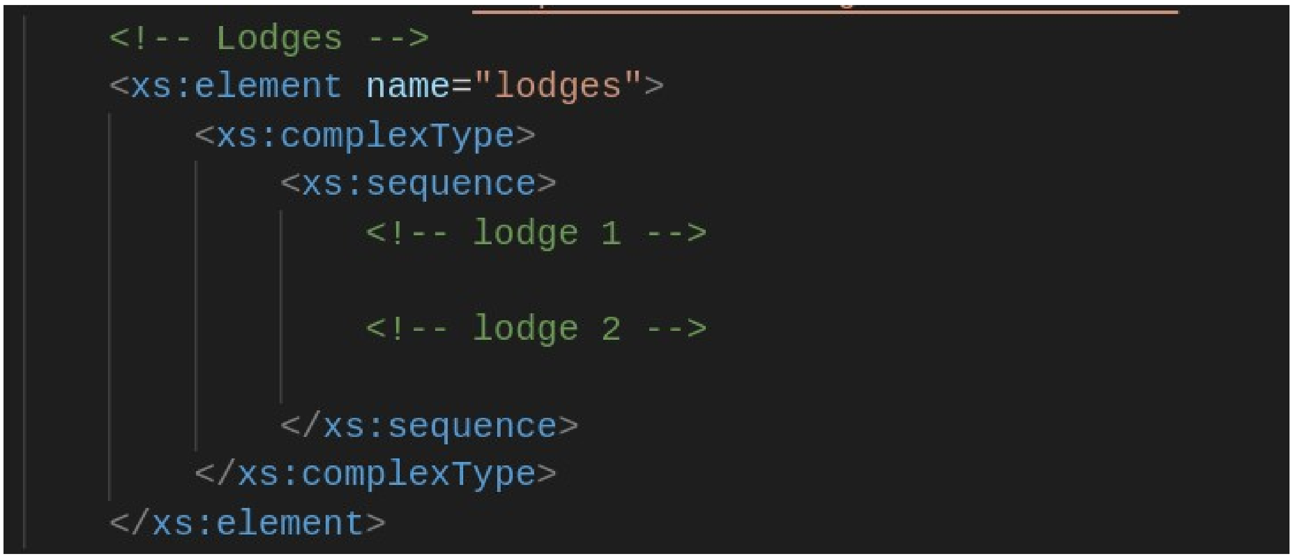
Create a root schema element and set the namespace attribute to the same URL as in your XML document:

- https://www.w3.org/2001/XMLSchema



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

<xs:schema xmlns:xs="https://www.w3.org/2001/XMLSchema">
    <!-- Lodges -->

</xs:schema>
```

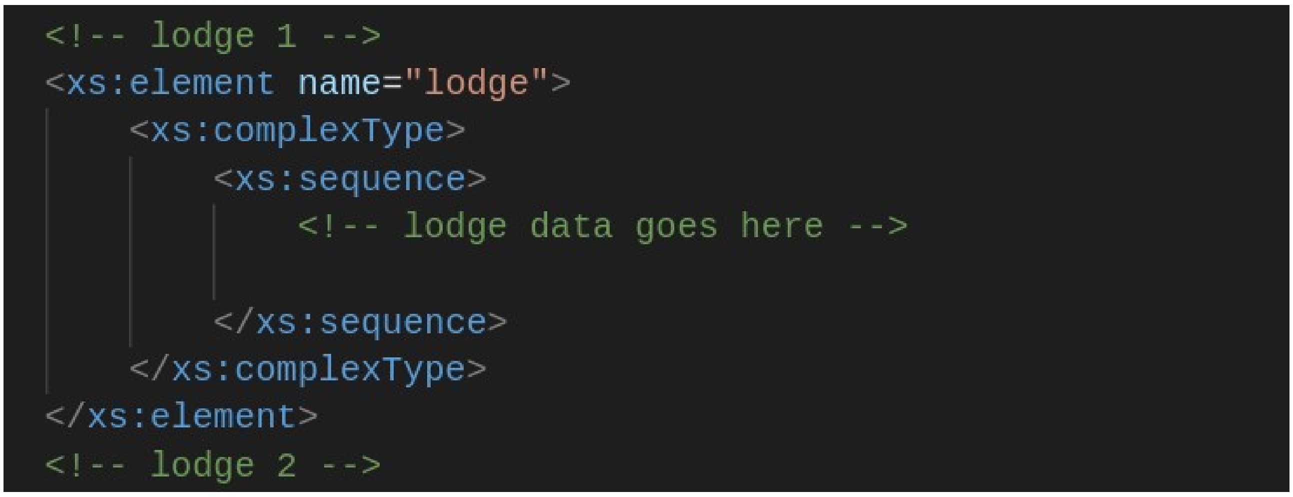
Inside the schema element, create a “**lodges**” element with a **complexType** and **sequence** nested elements inside:



```
    <!-- Lodges -->
    <xs:element name="lodges">
        <xs:complexType>
            <xs:sequence>
                <!-- lodge 1 -->

                <!-- lodge 2 -->

            </xs:sequence>
        </xs:complexType>
    </xs:element>
```

Examples of a range of XSD elements can be found here:
https://github.com/DavidAbram/xml-schema-cheat-sheet/blob/master/cheat-sheet.pdf

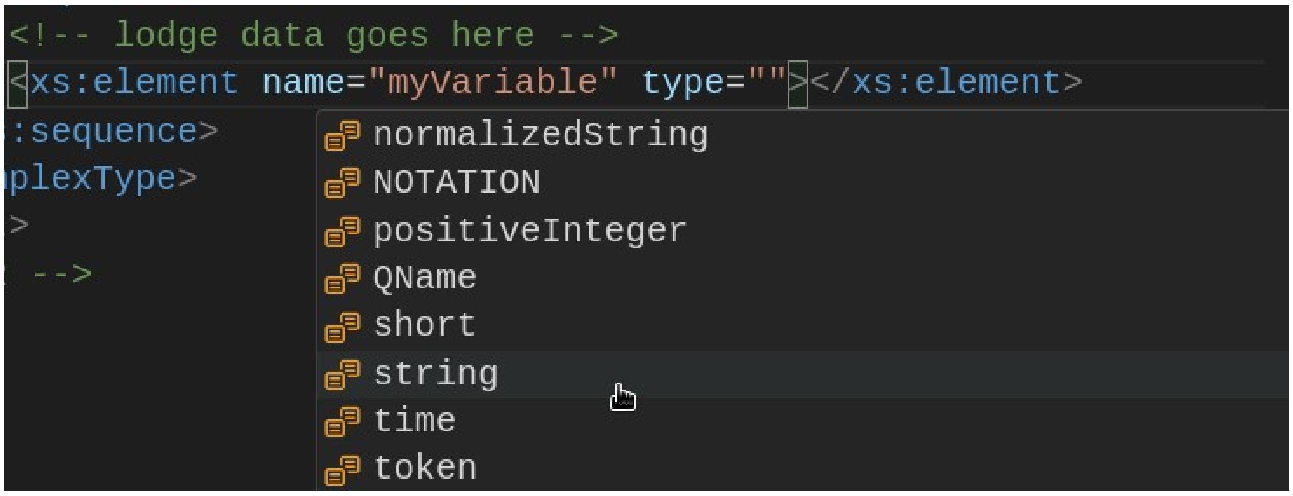
Inside the sequence element, create elements to represent **lodge 1** and its data:



```
<!-- lodge 1 -->
<xs:element name="lodge">
    <xs:complexType>
        <xs:sequence>
            <!-- lodge data goes here -->

        </xs:sequence>
    </xs:complexType>
</xs:element>
<!-- lodge 2 -->
```

When creating elements for single variables the **complexType** and **sequence** elements are **not required**. Instead, define an element with a **name** and data **type** that match the variables in your XML file.

NOTE: The RedHat extension will provide autofill prompts for the available data types. Select the most appropriate data type for your variable. This data type will be used to validate the value stored in the XML file.



```
<!-- lodge data goes here -->
<xs:element name="myVariable" type=""></xs:element>
:sequence>
plexType>
>
 -->
```

Once you have created the structure for lodge 1, it can be copy-pasted sequentially for each lodge in your XML file.

Here is an example of a completed XSD file portraying a books collection containing two books:

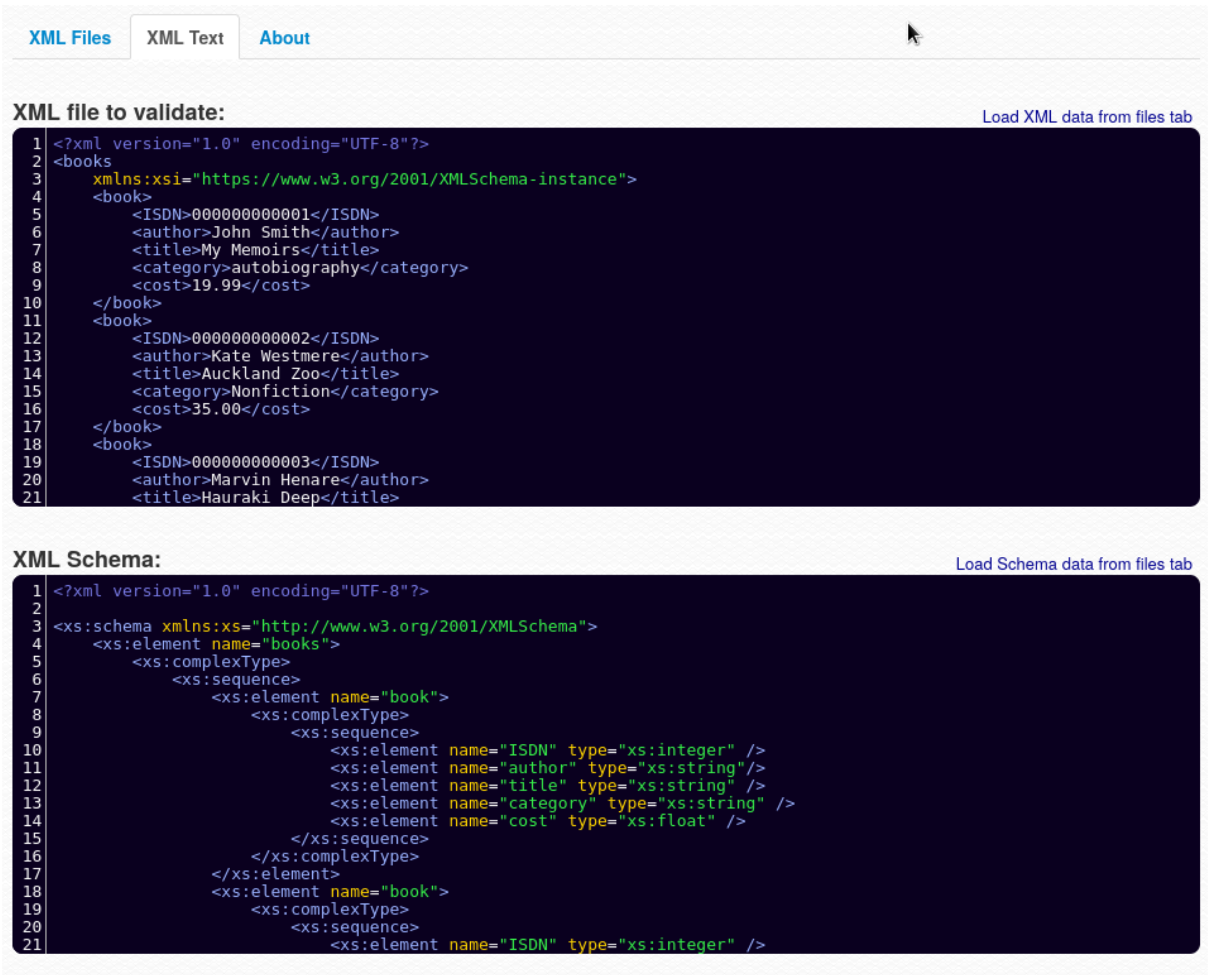
```xml
<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="books">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="book">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ISDN" type="xs:integer" />
                            <xs:element name="author" type="xs:string"/>
                            <xs:element name="title" type="xs:string" />
                            <xs:element name="category" type="xs:string" />
                            <xs:element name="cost" type="xs:float" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="book">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ISDN" type="xs:integer" />
                            <xs:element name="author" type="xs:string"/>
                            <xs:element name="title" type="xs:string" />
                            <xs:element name="category" type="xs:string" />
                            <xs:element name="cost" type="xs:float" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

continued

Validating your files

In a web browser go to: http://syssgx.github.io/xml.js/

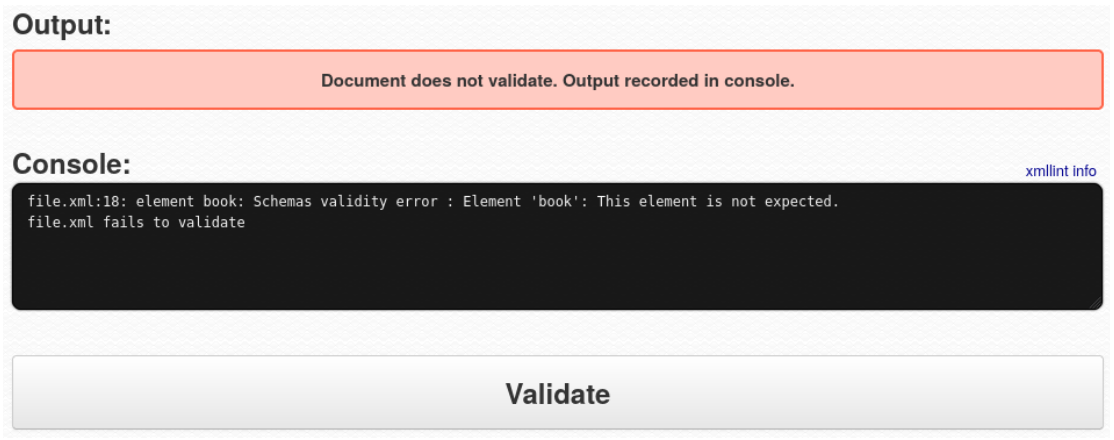
This website provides **validation** for XML and XSD file pairs. Drag & Drop your XML and XSD files on to the respective drop zones, or alternatively paste each files contents into the text fields under the XML Text tab:



```
<?xml version="1.0" encoding="UTF-8"?>
<books
    xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance">
    <book>
        <ISDN>000000000001</ISDN>
        <author>John Smith</author>
        <title>My Memoirs</title>
        <category>autobiography</category>
        <cost>19.99</cost>
    </book>
    <book>
        <ISDN>000000000002</ISDN>
        <author>Kate Westmere</author>
        <title>Auckland Zoo</title>
        <category>Nonfiction</category>
        <cost>35.00</cost>
    </book>
    <book>
        <ISDN>000000000003</ISDN>
        <author>Marvin Henare</author>
        <title>Hauraki Deep</title>
```

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="books">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="book">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ISDN" type="xs:integer" />
                            <xs:element name="author" type="xs:string"/>
                            <xs:element name="title" type="xs:string" />
                            <xs:element name="category" type="xs:string" />
                            <xs:element name="cost" type="xs:float" />
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="book">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ISDN" type="xs:integer" />
```

Once loaded, click the "**validate**" button below to check your files:



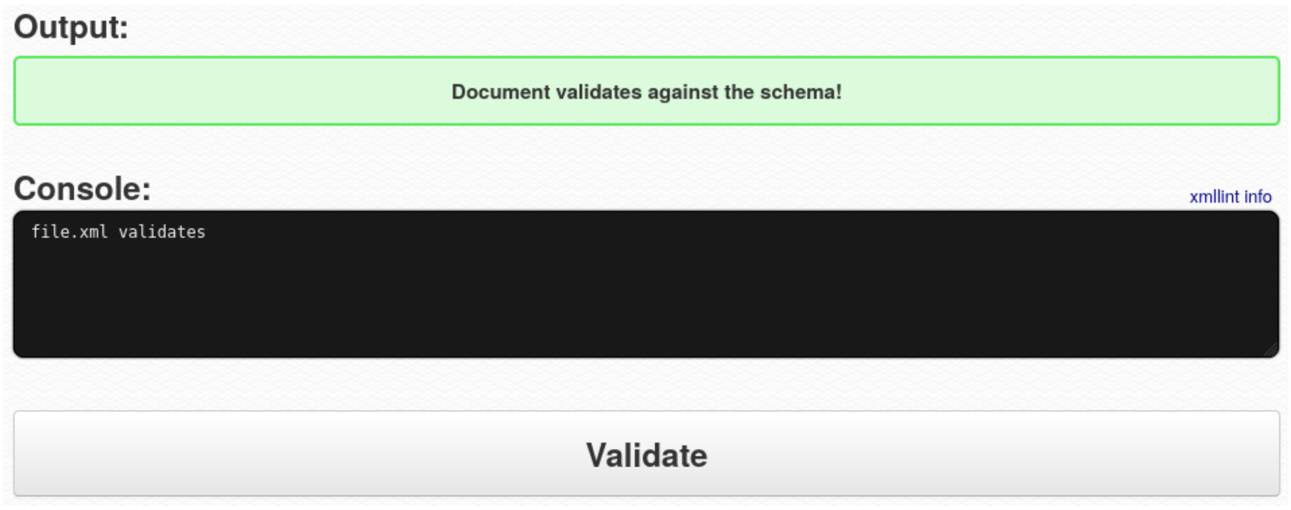
Document does not validate. Output recorded in console.

```
file.xml:18: element book: Schemas validity error : Element 'book': This element is not expected.
file.xml fails to validate
```

Resolving issues

If your files failed validation, the console will provide information indicating the source of the issue. In the above screenshot, the sample bookshop files failed to validate because the schema file only described two book elements inside the books list, but the XML file listed four books. The third book element on line 18 caused the failure.

To fix this example issue, the schema file needs to be updated to represent the expected four books:



Document validates against the schema!

```
file.xml validates
```

With the updates saved and added to the validation website, another check shows the files are valid and ready for use in an application.

Next steps

- Save your XML and XSD file contents as PDF documents with brief descriptive summary of the contents structure (copy the contents into a Word document, save as PDF). A template is provided on the last page of this exercise document.
- Add these PDF files to your Assessment 3 project.
- Add links from your project home page to the PDF documents.

Exercise complete.

Piha Lodges XML Design Document

ISCG6420 Assessment 3: Group Project

Summary

Provide a brief summary of the structure of your XML file. Explain what variables represent and their data types.

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<books
    xmlns:xsi="https://www.w3.org/2001/XMLSchema-instance">
    <book>
        <ISDN>000000000001</ISDN>
        <author>John Smith</author>
        <title>My Memoirs</title>
        <category>autobiography</category>
        <cost>19.99</cost>
    </book>
    <book>
        <ISDN>000000000002</ISDN>
        <author>Kate Westmere</author>
        <title>Auckland Zoo</title>
        <category>Nonfiction</category>
        <cost>35.00</cost>
    </book>
    <book>
        <ISDN>000000000003</ISDN>
        <author>Marvin Henare</author>
        <title>Hauraki Deep</title>
        <category>Fiction</category>
        <cost>28.99</cost>
    </book>
    <book>
        <ISDN>000000000004</ISDN>
        <author>Peter Long</author>
        <title>Vineyards of Waiheke</title>
        <category>Nonfiction</category>
        <cost>25.00</cost>
    </book>
</books>
```