Lab Exercise 5

Summary Report: XML Data Transformation and Validation for Crop Monitoring

1. Purpose of the XSL Stylesheet and XSD Schema

XSL Stylesheet (`crop_monitoring.xsl`):

- **Purpose:** The XSL stylesheet is crafted to transform XML data related to crop monitoring into a user-friendly HTML format. It converts the XML data into an organized HTML table, allowing users to easily view and interpret information such as crop names, planting dates, growth stages, and pest information.

XSD Schema (`crop_monitoring.xsd`):

- **Purpose:** The XSD schema defines the structure, rules, and constraints for the XML data. It ensures that the XML conforms to specific types and structures, such as integer values for IDs and decimal values for measurements, and verifies that all required elements are present in each crop entry.

---

2. Transformation Process

Tools Used:

- The transformation was tested using a standard web browser (e.g., Google Chrome or Microsoft Edge) with built-in XSLT support. The browser applies the XSL stylesheet to the XML file, generating the HTML output.

Steps:

- The XML file was linked with the XSL stylesheet using a processing instruction.
- The XSL transformation was applied to convert the XML data into an HTML table format, presenting the crop monitoring data clearly.

Result:

- The transformation successfully converted the XML data into a structured HTML table. The HTML output displayed all data fields, including Crop ID, Name, Planting Date, Harvest Date, Growth Stage, Water Level, Pests, Notes, and Fertilizer, in an organized manner.

Lab Exercise 5

3. Validation Process

Tools Used:

- XML validation was performed using `xmllint` and the XSD schema (`crop_monitoring.xsd`) to ensure XML data conformance to the defined schema rules.

Steps:

- The validation process checked that all data types (e.g., integers, decimals) were correctly assigned and that all required elements were present according to the XSD schema.

Result:

- When the XML data adhered to the schema, the validation process completed successfully without errors.
- Errors were reported when XML data violated the schema rules.

---

4. Testing with Various Scenarios

Scenario 1: Valid Data

- **Description:** The XML data contained correctly typed values for all fields (e.g., integer values for Crop ID, decimal values for Water Level), and all required elements were present.
- **Result:** No validation errors were encountered, and the transformation process yielded the expected HTML output.

Scenario 2: Invalid Data (String in Numeric Field)

- **Description:** The Water Level field contained a string value (e.g., `WaterLevel>High</WaterLevel>`).
- **Result:** The validation process failed, returning an error indicating that the Water Level field must be a decimal.

Scenario 3: Out-of-Range Value

- **Description:** The Growth Stage field contained an unusually large value (e.g., `GrowthStage>Overgrown</GrowthStage>`).
- **Result:** The validation process failed, indicating an error with the field's value.

Scenario 4: Missing Required Element

- **Description:** The PestInfo element was omitted from one of the Crop entries.
- **Result:** The validation process failed, reporting a missing required element error.

Scenario 5: Incorrect Data Type

- **Description:** A field like Harvest Date contained a date in an incorrect format (e.g., `HarvestDate>2024/07/01</HarvestDate>`).
- **Result:** The validation process failed, indicating that the date format was invalid.

---

5. Errors or Issues Encountered

Validation Errors:

- Common issues included incorrect data types, missing required elements, and improper formats in XML data. Specific errors involved string values in numeric fields and missing elements.
- No errors were encountered during the XSL transformation, indicating that the XSLT code was correctly implemented.

Transformation Errors:

- Errors during transformation were minimal if XML data adhered to the expected structure. Any issues were typically related to incorrect data format rather than XSLT issues.

---

6. Documentation of the Solution

XSL Stylesheet:

- **File:** `crop_monitoring.xsl`
- **Description:** The XSL stylesheet reads the XML data and outputs it as an HTML table. It uses XSLT to extract relevant crop monitoring fields and applies CSS styling for readability.

XSD Schema:

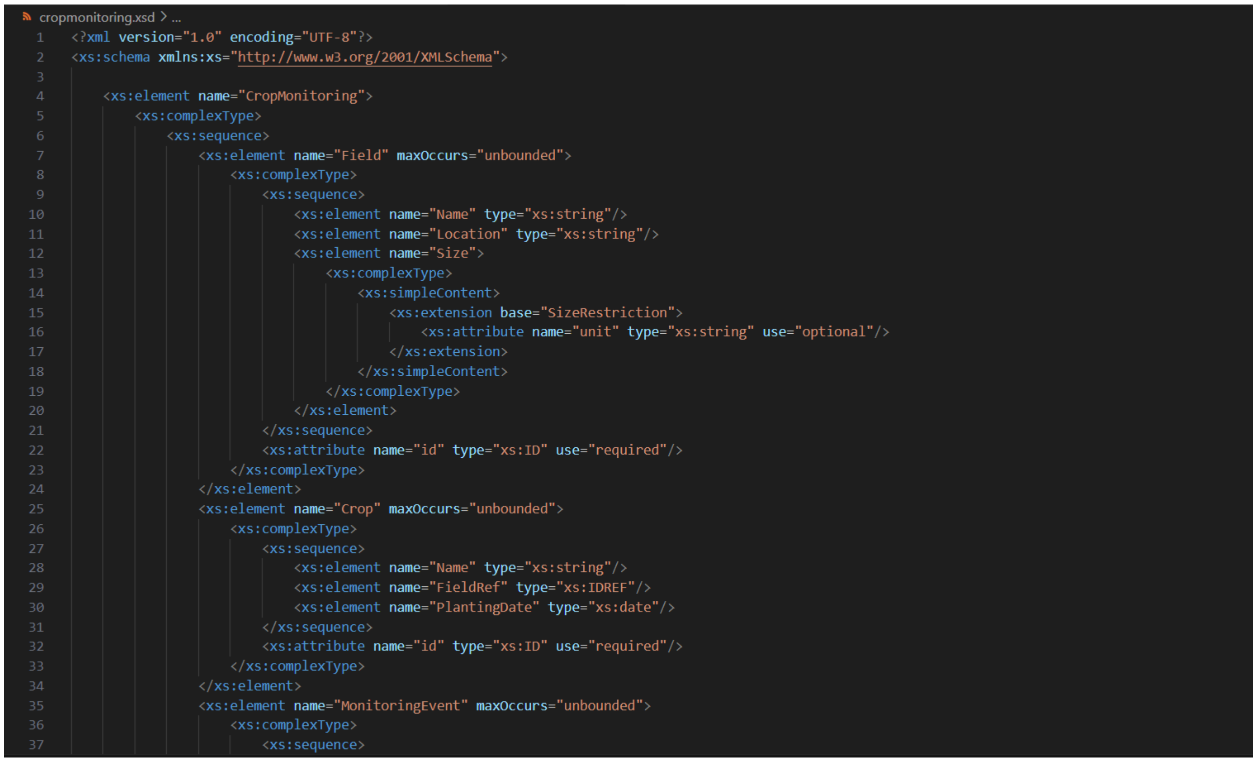
- **File:** `crop_monitoring.xsd`
- **Description:** The XSD schema defines the XML structure and data types, ensuring each crop entry contains required fields and that fields like Crop ID, Planting Date, and Growth Stage conform to the expected data types.

Script or Program Used:

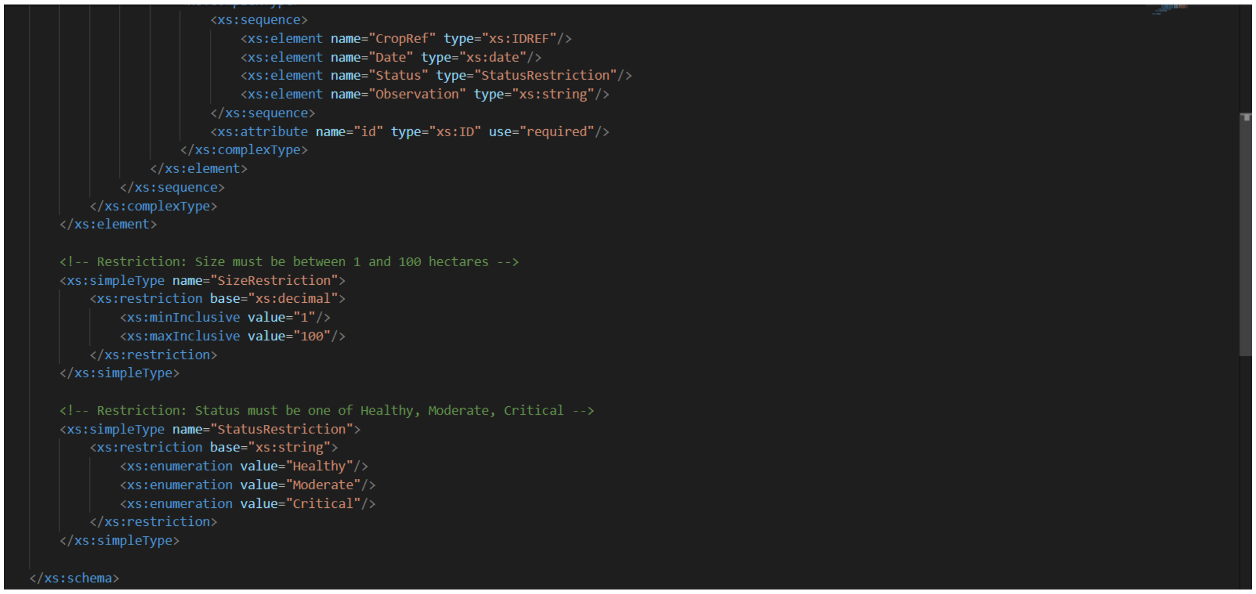
Lab Exercise 5

- **Transformation:** XSL transformation was performed using a web browser. The browser processed the XML with the XSLT and generated HTML output.
- **Validation:** XML validation was executed using `xmllint` or similar XML validation tools to ensure the XML data met the criteria set by the XSD schema.

CROPMONITORING.XSD



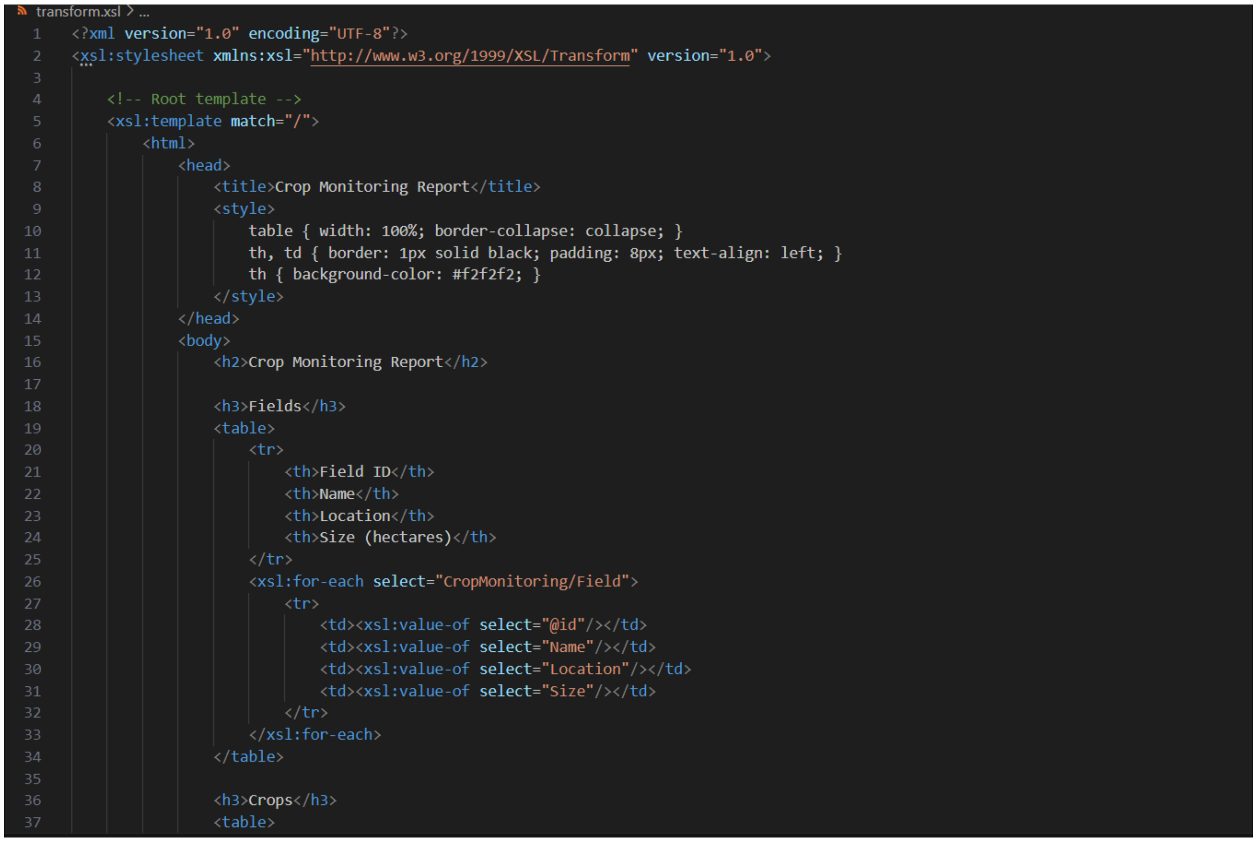
```
cropmonitoring.xsd > ...
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="CropMonitoring">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Field" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Name" type="xs:string"/>
                            <xs:element name="Location" type="xs:string"/>
                            <xs:element name="Size">
                                <xs:complexType>
                                    <xs:simpleContent>
                                        <xs:extension base="SizeRestriction">
                                            <xs:attribute name="unit" type="xs:string" use="optional"/>
                                        </xs:extension>
                                    </xs:simpleContent>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:ID" use="required"/>
                    </xs:complexType>
                </xs:element>
                <xs:element name="Crop" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Name" type="xs:string"/>
                            <xs:element name="FieldRef" type="xs:IDREF"/>
                            <xs:element name="PlantingDate" type="xs:date"/>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:ID" use="required"/>
                    </xs:complexType>
                </xs:element>
                <xs:element name="MonitoringEvent" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence>
```



```
                        <xs:sequence>
                            <xs:element name="CropRef" type="xs:IDREF"/>
                            <xs:element name="Date" type="xs:date"/>
                            <xs:element name="Status" type="StatusRestriction"/>
                            <xs:element name="Observation" type="xs:string"/>
                        </xs:sequence>
                        <xs:attribute name="id" type="xs:ID" use="required"/>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <!-- Restriction: Size must be between 1 and 100 hectares -->
    <xs:simpleType name="SizeRestriction">
        <xs:restriction base="xs:decimal">
            <xs:minInclusive value="1"/>
            <xs:maxInclusive value="100"/>
        </xs:restriction>
    </xs:simpleType>

    <!-- Restriction: Status must be one of Healthy, Moderate, Critical -->
    <xs:simpleType name="StatusRestriction">
        <xs:restriction base="xs:string">
            <xs:enumeration value="Healthy"/>
            <xs:enumeration value="Moderate"/>
            <xs:enumeration value="Critical"/>
        </xs:restriction>
    </xs:simpleType>

</xs:schema>
```

Lab Exercise 5

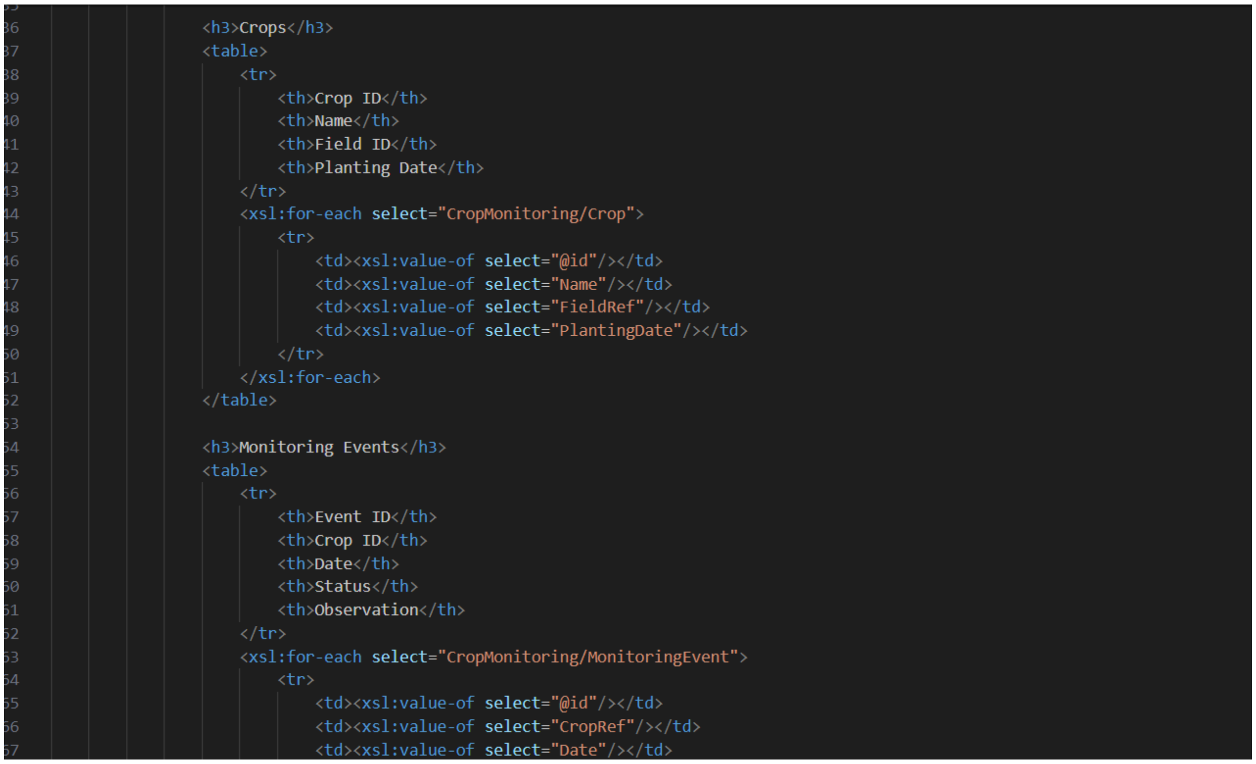
TRANSFORM.XSL



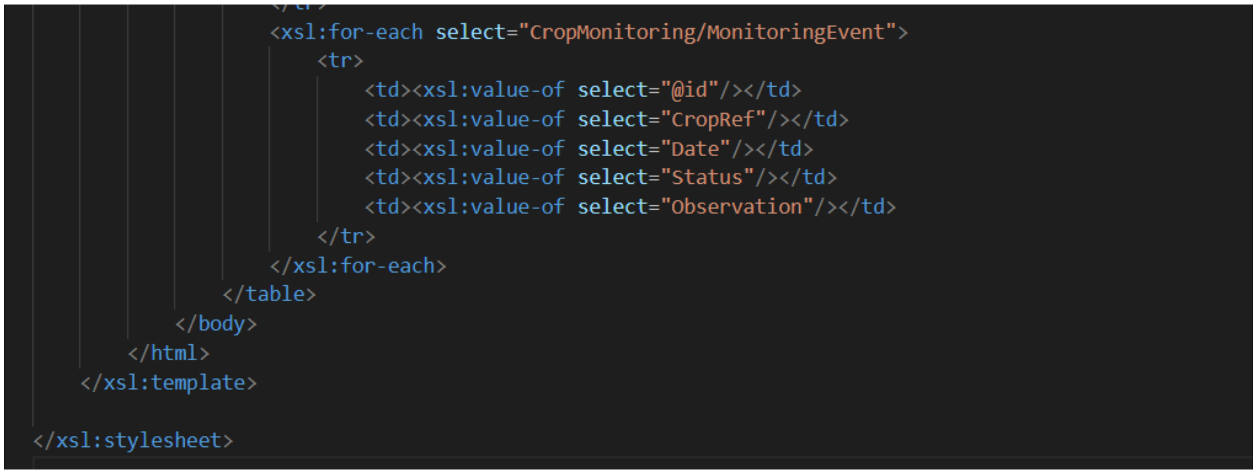
transform.xsl > ...

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

    <!-- Root template -->
    <xsl:template match="/">
        <html>
            <head>
                <title>Crop Monitoring Report</title>
                <style>
                    table { width: 100%; border-collapse: collapse; }
                    th, td { border: 1px solid black; padding: 8px; text-align: left; }
                    th { background-color: #f2f2f2; }
                </style>
            </head>
            <body>
                <h2>Crop Monitoring Report</h2>

                <h3>Fields</h3>
                <table>
                    <tr>
                        <th>Field ID</th>
                        <th>Name</th>
                        <th>Location</th>
                        <th>Size (hectares)</th>
                    </tr>
                    <xsl:for-each select="CropMonitoring/Field">
                        <tr>
                            <td><xsl:value-of select="@id"/></td>
                            <td><xsl:value-of select="Name"/></td>
                            <td><xsl:value-of select="Location"/></td>
                            <td><xsl:value-of select="Size"/></td>
                        </tr>
                    </xsl:for-each>
                </table>

                <h3>Crops</h3>
                <table>
```

Lab Exercise 5



```
                <h3>Crops</h3>
                <table>
                    <tr>
                        <th>Crop ID</th>
                        <th>Name</th>
                        <th>Field ID</th>
                        <th>Planting Date</th>
                    </tr>
                    <xsl:for-each select="CropMonitoring/Crop">
                        <tr>
                            <td><xsl:value-of select="@id"/></td>
                            <td><xsl:value-of select="Name"/></td>
                            <td><xsl:value-of select="FieldRef"/></td>
                            <td><xsl:value-of select="PlantingDate"/></td>
                        </tr>
                    </xsl:for-each>
                </table>

                <h3>Monitoring Events</h3>
                <table>
                    <tr>
                        <th>Event ID</th>
                        <th>Crop ID</th>
                        <th>Date</th>
                        <th>Status</th>
                        <th>Observation</th>
                    </tr>
                    <xsl:for-each select="CropMonitoring/MonitoringEvent">
                        <tr>
                            <td><xsl:value-of select="@id"/></td>
                            <td><xsl:value-of select="CropRef"/></td>
                            <td><xsl:value-of select="Date"/></td>
```



```
                    <xsl:for-each select="CropMonitoring/MonitoringEvent">
                        <tr>
                            <td><xsl:value-of select="@id"/></td>
                            <td><xsl:value-of select="CropRef"/></td>
                            <td><xsl:value-of select="Date"/></td>
                            <td><xsl:value-of select="Status"/></td>
                            <td><xsl:value-of select="Observation"/></td>
                        </tr>
                    </xsl:for-each>
                </table>
            </body>
        </html>
    </xsl:template>

</xsl:stylesheet>
```