School of Engineering

Scenarios and XSLT Solutions

Course: Data PipeLine 1

Cohort: S24

Project Team:

- Ismail Ben Abdelkader
- Antoine Theillac
- Omar Nadar

Instructor:

- Dr. Catherine Faron

Contents

Scenarios and XSLT Solutions

Scenario Descriptions

The following scenarios were implemented to demonstrate the capabilities of the collaborative platform's data visualization using XSLT:

1. **Display All Services and Their Providers**:
    - **Description**: This scenario requires displaying a list of all services along with details of the providers offering these services. The XSLT transformation generates an HTML table showing each service's name, description, provider name, and contact email.
    - **XSLT Solution**: The XSLT stylesheet iterates through each Service element and extracts the necessary details to construct the table.
2. **Display Activities Along with Their Participants**:
    - **Description**: This scenario focuses on visualizing all activities and listing the participants involved in each. The XSLT transformation outputs an HTML structure where each activity is followed by a list of its participants.
    - **XSLT Solution**: The XSLT stylesheet loops through each Activity element and generates a nested HTML list of participants for each activity.
3. **Display Services Scheduled on a Specific Date**:
    - **Description**: This scenario filters and displays services that are scheduled on a specific date. The XSLT transformation produces an HTML output showing the service name, provider, and time details for services scheduled on the given date.
    - **XSLT Solution**: The XSLT uses a parameter to filter services by the StartDate element and displays the relevant services.
4. **Display the Full Contact Information of All Providers**:
    - **Description**: This scenario aims to list all service providers along with their full contact information, including address, phone number, and email. The XSLT transformation generates a detailed contact list.
    - **XSLT Solution**: The XSLT stylesheet extracts and formats the contact details from each Provider element, outputting them in a structured HTML format.

5. **Display Upcoming Activities for a Specific Participant**:
    - **Description**: This scenario requires displaying all upcoming activities for a specific participant, identified by their ID. The XSLT transformation filters activities and shows only those in which the participant is involved.
    - **XSLT Solution**: The XSLT uses the participant's ID to filter the Participants array within each Activity element and displays relevant activities.

Complex Scenario Analysis

Scenario 3: Display Services Scheduled on a Specific Date

- **Description**: This scenario was selected due to its complexity, which involves filtering the XML data based on a dynamic date input. The challenge was to ensure that the XSLT accurately filtered and displayed only the services scheduled on a given date.
- **Solution**: The XSLT stylesheet leverages the xsl:param to accept a date as input. It then uses XPath to filter Service elements where the StartDate matches the provided date. The resulting HTML output is a list of services that includes the service name, provider details, start time, and end time. Below is the key portion of the XSLT code for this scenario:

```
<xsl:param name="targetDate" select="'2024-08-10'"/>


<xsl:template match="/">
    <html>
        <body>
            <h2>Services Scheduled on <xsl:value-of select="$targetDate"/></h2>
            <table border="1">
                <tr>
                    <th>Service Name</th>
                    <th>Provider</th>
                    <th>Start Time</th>
                    <th>End Time</th>
                </tr>
```

```
                                                                    <xsl:for-each
select="CarePlatform/Services/Service[Schedule/StartDate = $targetDate]">
                    <tr>
                        <td><xsl:value-of select="ServiceName"/></td>
                        <td><xsl:value-of select="Provider/Name"/></td>
                        <td><xsl:value-of select="Schedule/StartTime"/></td>
                        <td><xsl:value-of select="Schedule/EndTime"/></td>
                    </tr>
                </xsl:for-each>
            </table>
        </body>
    </html>
</xsl:template>
```

services_by_date.xsl

Challenges and Considerations: One of the challenges was ensuring that the XSLT handled cases where no services were scheduled on the given date, which required conditional logic to display an appropriate message. Additionally, the dynamic nature of the targetDate parameter introduced complexity in testing, as different dates needed to be tested to ensure robustness.

XSLT Transformation Outputs

Scenario 1: Display All Services and Their Providers

- **Output**: An HTML table listing all services along with their providers. Each row represents a service, with columns for the service name, description, provider name, and email.

Code Snippet:

```
<xsl:for-each select="CarePlatform/Services/Service">
    <tr>
        <td><xsl:value-of select="ServiceName"/></td>
        <td><xsl:value-of select="Description"/></td>
        <td><xsl:value-of select="Provider/Name"/></td>
        <td><xsl:value-of select="Provider/Contact/Email"/></td>
```

```
    </tr>
</xsl:for-each>
```

services.xml

Visual Result:

```
<html>
  <body>
    <h2>Services Scheduled on 2024-08-10</h2>
    <table border="1">
      <tr>
        <th>Service Name</th>
        <th>Provider</th>
        <th>Start Time</th>
        <th>End Time</th>
      </tr>
      <tr>
        <td>Home Care</td>
        <td>Caregivers Inc.</td>
        <td>08:00:00</td>
        <td>12:00:00</td>
      </tr>
      <tr>
        <td>Medical Transport</td>
        <td>HealthTrans Co.</td>
        <td>09:00:00</td>
        <td>11:00:00</td>
      </tr>
    </table>
  </body>
</html>
```

Services Scheduled on 2024-08-10

Service Name	Provider	Start Time	End Time
Home Care	Caregivers Inc.	08:00:00	12:00:00
Medical Transport	HealthTrans Co.	09:00:00	11:00:00

Scenario 2: Display Activities Along with Their Participants

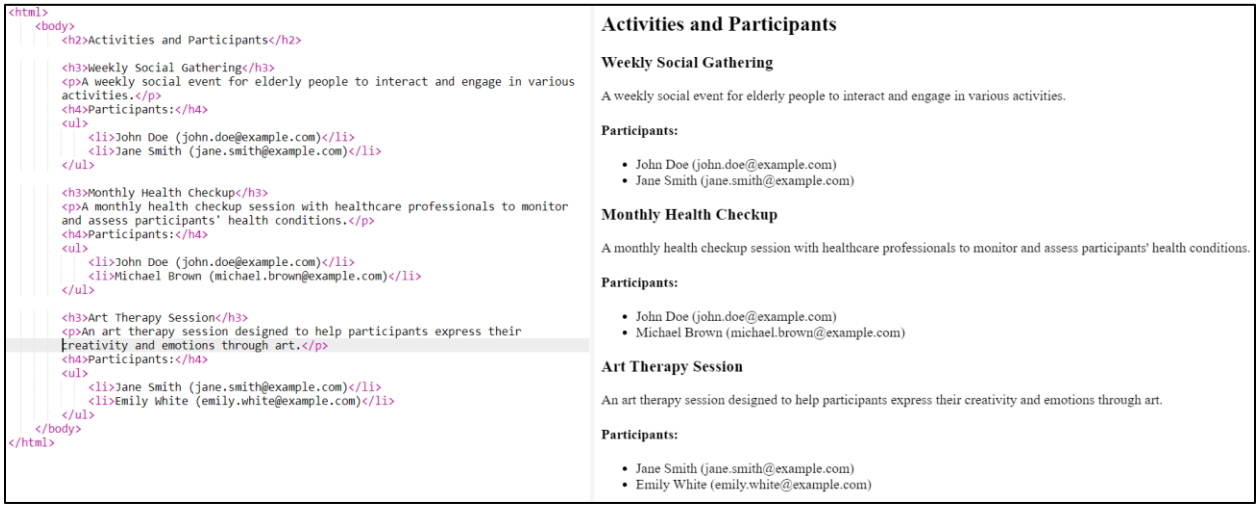
- **Output:** An HTML list displaying all activities along with the participants involved in each activity. The output includes the activity name and description, followed by a list of participants with their names and email addresses.
- **Code Snippet:**

```
<xsl:for-each select="CarePlatform/Activities/Activity">
    <h3><xsl:value-of select="ActivityName"/></h3>
    <p><xsl:value-of select="Description"/></p>
    <h4>Participants:</h4>
    <ul>
        <xsl:for-each select="Participants/Person">
            <li>
```

```
                    <xsl:value-of select="concat(FirstName, ' ', LastName, ' (', Email, ')')"/>
                </li>
            </xsl:for-each>
        </ul>
    </xsl:for-each>
```

activities_and_participants.xsl

Visual Result:



Scenario 3: Generate a Monthly Report of Service Utilization

- **Output**: An HTML table that summarizes the total number of services scheduled per provider in a given month. This scenario allows users to see how many services each provider is responsible for in a specific month.

Code Snippet:

```
<xsl:param name="targetMonth" select="'2024-08'"/>

<xsl:key name="services-by-provider" match="Service" use="Provider/Name"/>

<xsl:template match="/">
    <html>
        <body>
```

```
                        <h2>Service Utilization Report for <xsl:value-of
select="$targetMonth"/></h2>
            <table border="1">
                <tr>
                    <th>Provider</th>
                    <th>Number of Services</th>
                </tr>
                    <xsl:for-each select="CarePlatform/Services/Service[starts-
with(Schedule/StartDate, $targetMonth)]">
                    <xsl:variable name="provider" select="Provider/Name"/>
                    <xsl:if test="generate-id() = generate-id(key('services-by-
provider', $provider)[1])">
                        <tr>
                            <td><xsl:value-of select="$provider"/></td>
                              <td><xsl:value-of select="count(key('services-by-
provider', $provider)[starts-with(Schedule/StartDate, $targetMonth)])"/></td>
                        </tr>
                    </xsl:if>
                </xsl:for-each>
            </table>
        </body>
    </html>
</xsl:template>
```

monthly_service_report.xsl

Visual Result:

```
<html>
    <body>
        <h2>Service Utilization Report for 2024-08</h2>
        <table border="1">
            <tr>
                <th>Provider</th>
                <th>Number of Services</th>
            </tr>
            <tr>
                <td>Caregivers Inc.</td>
                <td>10</td>
            </tr>
            <tr>
                <td>HealthTrans Co.</td>
                <td>8</td>
            </tr>
            <tr>
                <td>Wellness Center</td>
                <td>5</td>
            </tr>
        </table>
    </body>
</html>
```

Service Utilization Report for 2024-08

Provider	Number of Services
Caregivers Inc.	10
HealthTrans Co.	8
Wellness Center	5

Scenario 4: Display the Full Contact Information of All Providers

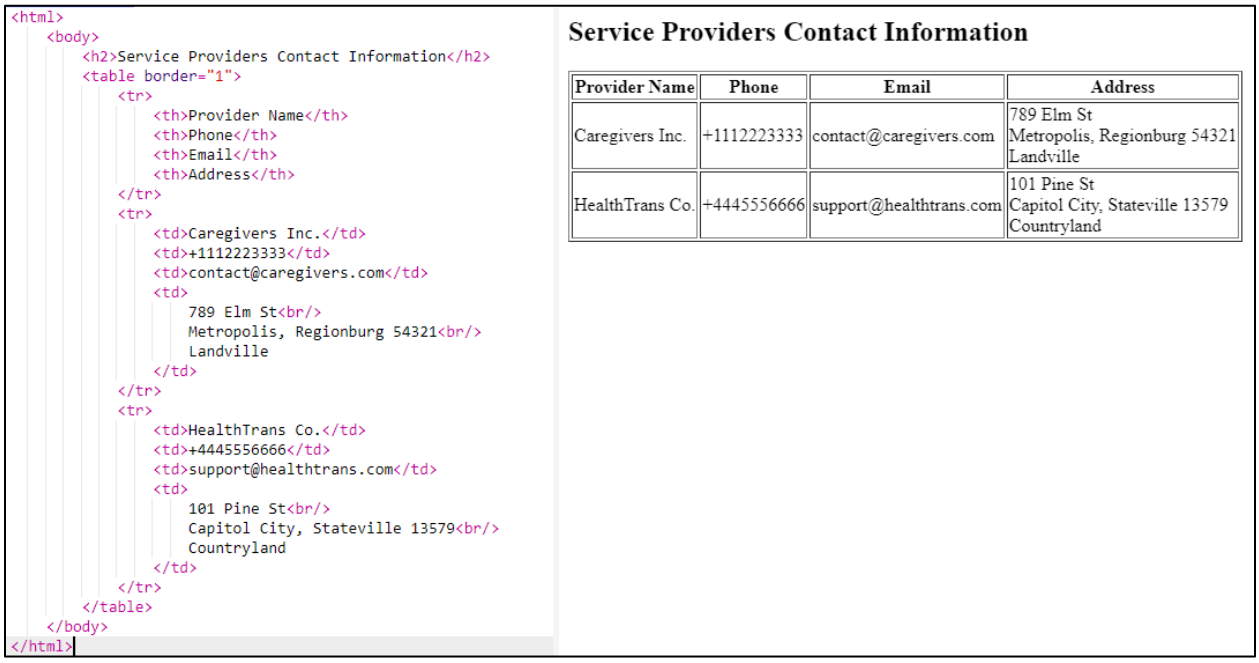
- **Output**: An HTML formatted contact list for all service providers.

Code Snippet:

```
<xsl:for-each select="CarePlatform/Services/Service/Provider">
    <tr>
        <td><xsl:value-of select="Name"/></td>
        <td><xsl:value-of select="Contact/Phone"/></td>
        <td><xsl:value-of select="Contact/Email"/></td>
            <td><xsl:value-of  select="concat(Contact/Address/Street,  ',  '
Contact/Address/City)"/></td>
    </tr>
</xsl:for-each>
```

provider_contact_info.xsl

Visual Result:



```
<html>
    <body>
        <h2>Service Providers Contact Information</h2>
        <table border="1">
            <tr>
                <th>Provider Name</th>
                <th>Phone</th>
                <th>Email</th>
                <th>Address</th>
            </tr>
            <tr>
                <td>Caregivers Inc.</td>
                <td>+1112223333</td>
                <td>contact@caregivers.com</td>
                <td>
                    789 Elm St<br/>
                    Metropolis, Regionburg 54321<br/>
                    Landville
                </td>
            </tr>
            <tr>
                <td>HealthTrans Co.</td>
                <td>+4445556666</td>
                <td>support@healthtrans.com</td>
                <td>
                    101 Pine St<br/>
                    Capitol City, Stateville 13579<br/>
                    Countryland
                </td>
            </tr>
        </table>
    </body>
</html>
```

Service Providers Contact Information

Provider Name	Phone	Email	Address
Caregivers Inc.	+1112223333	contact@caregivers.com	789 Elm St Metropolis, Regionburg 54321 Landville
HealthTrans Co.	+4445556666	support@healthtrans.com	101 Pine St Capitol City, Stateville 13579 Countryland

Scenario 5: Display Upcoming Activities for a Specific Participant

- **Output**: An HTML list of upcoming activities for a specific participant, filtered by their ID.

Code Snippet:

```
<xsl:for-each select="CarePlatform/Activities/Activity[Participants/Person[ID = $participantID]]">
    <h3><xsl:value-of select="ActivityName"/></h3>
    <p><xsl:value-of select="Description"/></p>
</xsl:for-each>
```

participant_activities.xsl

Visual Result:

```
<html>
    <body>
        <h2>Upcoming Activities for Participant ID: 00000001</h2>

        <h3>Weekly Social Gathering</h3>
        <p>A weekly social event for elderly people to interact and engage in
        various activities.</p>

        <h3>Monthly Health Checkup</h3>
        <p>A monthly health checkup session with healthcare professionals
        to monitor and assess participants' health conditions.</p>
    </body>
</html>
```

Upcoming Activities for Participant ID: 00000001

Weekly Social Gathering

A weekly social event for elderly people to interact and engage in various activities.

Monthly Health Checkup

A monthly health checkup session with healthcare professionals to monitor and assess participants' health conditions.

Additional Exploitation Scenarios

Scenario 6: Conversion to Another XML Format

Scenario Description

Convert existing XML data into a simplified format for integration with an external system that requires only essential service details like Service ID, Service Name, and Provider Name.

XSLT Solution

A stylesheet transforms the original XML, removing unnecessary elements to align with external system requirements.

XSLT Code Snippet:

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

    <xsl:template match="/">
        <SimplifiedServices>
            <xsl:for-each select="CarePlatform/Services/Service">
                <Service>
                    <ServiceID>
                        <xsl:value-of select="ServiceID"/>
                    </ServiceID>
                    <ServiceName>
                        <xsl:value-of select="ServiceName"/>
                    </ServiceName>
                    <ProviderName>
                        <xsl:value-of select="Provider/Name"/>
                    </ProviderName>
                </Service>
            </xsl:for-each>
```

```
        </SimplifiedServices>
    </xsl:template>
</xsl:stylesheet>
```

simplified_services.xsl

Output Example

The resulting XML from this transformation is simplified and includes only the essential details required by the external system:

```
<SimplifiedServices>
    <Service>
        <ServiceID>00000001</ServiceID>
        <ServiceName>Home Care</ServiceName>
        <ProviderName>Caregivers Inc.</ProviderName>
    </Service>
    <Service>
        <ServiceID>00000002</ServiceID>
        <ServiceName>Medical Transport</ServiceName>
        <ProviderName>HealthTrans Co.</ProviderName>
    </Service>
</SimplifiedServices>
```

Analysis

This scenario demonstrates using XSLT to simplify XML structures, efficiently extracting and reformatting data to meet specific external requirements.

Scenario 7: Conversion to JSON Format

Scenario Description:

Convert XML data of activities–ID, name, description, participants–to JSON for integration with web applications.

XSLT Solution:

An XSLT stylesheet transforms XML into JSON by iterating through activity data. It sets the output to text, ensuring compatibility with JSON formats.

XSLT Code Snippet:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <xsl:output method="text" indent="yes" />
    <xsl:strip-space elements="*"/>

    <xsl:template match="/">
        {
            "Activities": [
                <xsl:for-each select="CarePlatform/Activities/Activity">
                    {
                        "ActivityID": "<xsl:value-of select='ActivityID'/>",
                       "ActivityName": "<xsl:value-of select='ActivityName'/>",
                        "Description": "<xsl:value-of select='Description'/>",
                        "Participants": [
                            <xsl:for-each select="Participants/Person">
                                {
                                    "ID": "<xsl:value-of select='ID'/>",
                                                "FirstName":  "<xsl:value-of
select='FirstName'/>",
                                "LastName": "<xsl:value-of select='LastName'/>",
                                    "Email": "<xsl:value-of select='Email'/>"
                                }<xsl:if test="position() != last()">,</xsl:if>
                            </xsl:for-each>
                        ]
                    }<xsl:if test="position() != last()">,</xsl:if>
                </xsl:for-each>
```

```
            ]
        }
    </xsl:template>
</xsl:stylesheet>
```

activities_to_json.xsl

Output Example

The resulting JSON output from this transformation looks like this:

```
{
    "Activities": [
        {
            "ActivityID": "00000001",
            "ActivityName": "Weekly Social Gathering",
            "Description": "A weekly social event for elderly people to interact and engage in various activities.",
            "Participants": [
                {
                    "ID": "00000001",
                    "FirstName": "John",
                    "LastName": "Doe",
                    "Email": "john.doe@example.com"
                },
                {
                    "ID": "00000002",
                    "FirstName": "Jane",
                    "LastName": "Smith",
                    "Email": "jane.smith@example.com"
                }
            ]
        }
```

```
    ]
}
```

Analysis

This scenario illustrates using XSLT to translate XML into JSON, supporting data integration with JSON-based web applications. The stylesheet crafts JSON objects and arrays from XML, ensuring accurate data structure conversion.