School of Engineering

Code : Collaborative Platform

Course: Data PipeLine 1

Cohort: S24

Project Team:

- Ismail Ben Abdelkader
- Antoine Theillac
- Omar Nadar

Instructor:

- Dr. Catherine Faron

Contents

Appendices

Appendix A: Complete XML Schema.

Below is the complete XML Schema (XSD) that was developed for the project. This schema defines the structure of the XML documents used in the collaborative platform for social and medical care for disabled and elderly individuals.

```
Appendices
Appendix A Complete XML Schema.
Below is the complete XML Schema (XSD) that was developed for the project. This
schema defines the structure of the XML documents used in the collaborative
platform for social and medical care for disabled and elderly individuals.
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">

    <!-- Definition of simple types -->
    <xs:simpleType name="IDType">
        <xs:restriction base="xs:string">
            <xs:pattern value="\d{8}"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="DateType">
        <xs:restriction base="xs:date"/>
    </xs:simpleType>

    <xs:simpleType name="TimeType">
        <xs:restriction base="xs:time"/>
    </xs:simpleType>

    <xs:simpleType name="EmailType">
        <xs:restriction base="xs:string">
           <xs:pattern value="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}"/>
        </xs:restriction>
    </xs:simpleType>

    <!-- Definition of complex types -->
    <xs:complexType name="PersonType">
        <xs:sequence>
            <xs:element name="ID" type="IDType"/>
            <xs:element name="FirstName" type="xs:string"/>
```

```
            <xs:element name="LastName" type="xs:string"/>
            <xs:element name="DOB" type="DateType"/>
            <xs:element name="Email" type="EmailType" minOccurs="0"/>
            <xs:element name="Phone" type="xs:string" minOccurs="0"/>
            <xs:element name="Address" type="AddressType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="AddressType">
        <xs:sequence>
            <xs:element name="Street" type="xs:string"/>
            <xs:element name="City" type="xs:string"/>
            <xs:element name="State" type="xs:string"/>
            <xs:element name="PostalCode" type="xs:string"/>
            <xs:element name="Country" type="xs:string"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ServiceType">
        <xs:sequence>
            <xs:element name="ServiceID" type="IDType"/>
            <xs:element name="ServiceName" type="xs:string"/>
            <xs:element name="Description" type="xs:string"/>
            <xs:element name="Provider" type="ProviderType"/>
            <xs:element name="Schedule" type="ScheduleType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ProviderType">
        <xs:sequence>
            <xs:element name="ProviderID" type="IDType"/>
            <xs:element name="Name" type="xs:string"/>
            <xs:element name="Contact" type="ContactType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ContactType">
        <xs:sequence>
            <xs:element name="Phone" type="xs:string"/>
            <xs:element name="Email" type="EmailType"/>
            <xs:element name="Address" type="AddressType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ScheduleType">
```

```
        <xs:sequence>
            <xs:element name="StartDate" type="DateType"/>
            <xs:element name="EndDate" type="DateType"/>
            <xs:element name="StartTime" type="TimeType"/>
            <xs:element name="EndTime" type="TimeType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ActivityType">
        <xs:sequence>
            <xs:element name="ActivityID" type="IDType"/>
            <xs:element name="ActivityName" type="xs:string"/>
            <xs:element name="Description" type="xs:string"/>
            <xs:element name="Participants" type="ParticipantsType"/>
            <xs:element name="Schedule" type="ScheduleType"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ParticipantsType">
        <xs:sequence>
            <xs:element name="Person" type="PersonType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <!-- Root element -->
    <xs:element name="CarePlatform">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Persons" type="PersonsType"/>
                <xs:element name="Services" type="ServicesType"/>
                <xs:element name="Activities" type="ActivitiesType"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <!-- Aggregation of complex types -->
    <xs:complexType name="PersonsType">
        <xs:sequence>
            <xs:element name="Person" type="PersonType" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

    <xs:complexType name="ServicesType">
        <xs:sequence>
           <xs:element name="Service" type="ServiceType" maxOccurs="unbounded"/>
```

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

    <xs:complexType name="ActivitiesType">
        <xs:sequence>
                        <xs:element    name="Activity"    type="ActivityType"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>

</xs:schema>
```

care_platform_schema.xsd

Explanation

- **PersonType**: Defines the structure for individual person records, including ID, name, date of birth, contact details, and address.
- **ServiceType**: Specifies the details of services offered on the platform, including service ID, name, description, provider information, and schedule.
- **ActivityType**: Represents activities organized on the platform, including activity ID, name, description, participant list, and schedule.
- **ProviderType**: Describes the providers offering services, including their contact information.
- **ScheduleType**: Provides the timing details for services and activities.

Appendix B: Sample XML Database.

Below is a sample XML database that adheres to the XML Schema outlined in Appendix A. This sample data is representative of the information managed by the collaborative platform for social and medical care for disabled and elderly individuals.

```
<?xml version="1.0" encoding="UTF-8"?>
<CarePlatform xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:noNamespaceSchemaLocation="care_platform_schema.xsd">
    <!-- List of Persons -->
    <Persons>
        <Person>
            <ID>00000001</ID>
            <FirstName>John</FirstName>
            <LastName>Doe</LastName>
```

```
            <DOB>1945-05-12</DOB>
            <Email>john.doe@example.com</Email>
            <Phone>+1234567890</Phone>
            <Address>
                <Street>123 Main St</Street>
                <City>Anytown</City>
                <State>Stateville</State>
                <PostalCode>12345</PostalCode>
                <Country>Countryland</Country>
            </Address>
        </Person>
        <Person>
            <ID>00000002</ID>
            <FirstName>Jane</FirstName>
            <LastName>Smith</LastName>
            <DOB>1950-09-22</DOB>
            <Email>jane.smith@example.com</Email>
            <Phone>+0987654321</Phone>
            <Address>
                <Street>456 Oak Ave</Street>
                <City>Sometown</City>
                <State>Regionburg</State>
                <PostalCode>67890</PostalCode>
                <Country>Landville</Country>
            </Address>
        </Person>
        <!-- Additional Persons -->
        <Person>
            <ID>00000003</ID>
            <FirstName>Alice</FirstName>
            <LastName>Johnson</LastName>
            <DOB>1938-11-16</DOB>
            <Email>alice.johnson@example.com</Email>
            <Phone>+1029384756</Phone>
            <Address>
                <Street>789 Cedar Ln</Street>
                <City>Laketown</City>
                <State>Lakeview</State>
                <PostalCode>45678</PostalCode>
                <Country>Countryland</Country>
            </Address>
        </Person>
    </Persons>

    <!-- List of Services -->
```

```
    <Services>
        <Service>
            <ServiceID>00000001</ServiceID>
            <ServiceName>Home Care</ServiceName>
            <Description>Personal assistance and home care services for daily
needs.</Description>
            <Provider>
                <ProviderID>00000001</ProviderID>
                <Name>Caregivers Inc.</Name>
                <Contact>
                    <Phone>+1112223333</Phone>
                    <Email>contact@caregivers.com</Email>
                    <Address>
                        <Street>789 Elm St</Street>
                        <City>Metropolis</City>
                        <State>Regionburg</State>
                        <PostalCode>54321</PostalCode>
                        <Country>Landville</Country>
                    </Address>
                </Contact>
            </Provider>
            <Schedule>
                <StartDate>2024-08-01</StartDate>
                <EndDate>2024-12-31</EndDate>
                <StartTime>08:00:00</StartTime>
                <EndTime>18:00:00</EndTime>
            </Schedule>
        </Service>
        <Service>
            <ServiceID>00000002</ServiceID>
            <ServiceName>Medical Transport</ServiceName>
            <Description>Transportation services for medical appointments and
hospital visits.</Description>
            <Provider>
                <ProviderID>00000002</ProviderID>
                <Name>HealthTrans Co.</Name>
                <Contact>
                    <Phone>+4445556666</Phone>
                    <Email>support@healthtrans.com</Email>
                    <Address>
                        <Street>101 Pine St</Street>
                        <City>Capitol City</City>
                        <State>Stateville</State>
                        <PostalCode>13579</PostalCode>
                        <Country>Countryland</Country>
```

```
                    </Address>
                </Contact>
            </Provider>
            <Schedule>
                <StartDate>2024-08-10</StartDate>
                <EndDate>2024-12-31</EndDate>
                <StartTime>07:00:00</StartTime>
                <EndTime>20:00:00</EndTime>
            </Schedule>
        </Service>
        <!-- Additional Services -->
        <Service>
            <ServiceID>00000003</ServiceID>
            <ServiceName>Meal Delivery</ServiceName>
            <Description>Nutritionally balanced meal delivery for elderly and
disabled.</Description>
            <Provider>
                <ProviderID>00000003</ProviderID>
                <Name>FoodCare Services</Name>
                <Contact>
                    <Phone>+1597532846</Phone>
                    <Email>info@foodcare.com</Email>
                    <Address>
                        <Street>456 Broad Way</Street>
                        <City>Grandville</City>
                        <State>Stateville</State>
                        <PostalCode>56789</PostalCode>
                        <Country>Countryland</Country>
                    </Address>
                </Contact>
            </Provider>
            <Schedule>
                <StartDate>2024-08-01</StartDate>
                <EndDate>2024-12-31</EndDate>
                <StartTime>10:00:00</StartTime>
                <EndTime>14:00:00</EndTime>
            </Schedule>
        </Service>
    </Services>

    <!-- List of Activities -->
    <Activities>
        <Activity>
            <ActivityID>00000001</ActivityID>
            <ActivityName>Weekly Social Gathering</ActivityName>
```

```
            <Description>A weekly social event for elderly people to interact
and engage in various activities.</Description>
            <Participants>
                <Person>
                    <ID>00000001</ID>
                    <FirstName>John</FirstName>
                    <LastName>Doe</LastName>
                    <DOB>1945-05-12</DOB>
                    <Email>john.doe@example.com</Email>
                    <Phone>+1234567890</Phone>
                    <Address>
                        <Street>123 Main St</Street>
                        <City>Anytown</City>
                        <State>Stateville</State>
                        <PostalCode>12345</PostalCode>
                        <Country>Countryland</Country>
                    </Address>
                </Person>
                <Person>
                    <ID>00000003</ID>
                    <FirstName>Alice</FirstName>
                    <LastName>Johnson</LastName>
                    <DOB>1938-11-16</DOB>
                    <Email>alice.johnson@example.com</Email>
                    <Phone>+1029384756</Phone>
                    <Address>
                        <Street>789 Cedar Ln</Street>
                        <City>Laketown</City>
                        <State>Lakeview</State>
                        <PostalCode>45678</PostalCode>
                        <Country>Countryland</Country>
                    </Address>
                </Person>
            </Participants>
            <Schedule>
                <StartDate>2024-08-10</StartDate>
                <EndDate>2024-12-31</EndDate>
                <StartTime>10:00:00</StartTime>
                <EndTime>14:00:00</EndTime>
            Schedule>
        </Activity>
    </Activities>
</CarePlatform>
```

complete_care_platform_schema.xsd

Explanation

- **Persons**: Contains a list of individuals involved in the platform, with detailed information including ID, name, contact details, and address.
- **Services**: Lists the services offered on the platform, each including service ID, name, description, provider details, and scheduling information.
- **Activities**: Includes activities organized for the platform's participants, detailing the activity ID, name, description, participants, and schedule.

Appendix C: XSLT Files with Detailed Comments.

This appendix includes the XSLT files developed for the project, each accompanied by detailed comments to explain their functionality, structure, and logic. These XSLT stylesheets are used to transform XML data into various formats, such as HTML and JSON, to meet specific visualization and data export needs.

XSLT File 1: Display All Services and Their Providers

Purpose: This XSLT stylesheet is designed to display a list of all services along with the details of the providers offering these services. The output is formatted as an HTML table.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <!-- Output method set to HTML for browser compatibility -->
    <xsl:output method="html" indent="yes"/>

    <!-- Root template matches the entire XML document -->
    <xsl:template match="/">
        <html>
            <body>
                <h2>List of Services and Providers</h2>
                <!-- Begin HTML table -->
                <table border="1">
                    <tr>
                        <th>Service Name</th>
                        <th>Description</th>
                        <th>Provider Name</th>
                        <th>Contact Email</th>
                    </tr>
                    <!-- Iterate over each Service element -->
                    <xsl:for-each select="CarePlatform/Services/Service">
                        <tr>
```

```
                            <!-- Extract and display the service name -->
                            <td><xsl:value-of select="ServiceName"/></td>
                            <!-- Extract and display the service description --
>
                            <td><xsl:value-of select="Description"/></td>
                            <!-- Extract and display the provider's name -->
                            <td><xsl:value-of select="Provider/Name"/></td>
                            <!-- Extract and display the provider's contact
email -->
                            <td><xsl:value-of
select="Provider/Contact/Email"/></td>
                        </tr>
                    </xsl:for-each>
                </table>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

services_and_providers.xsl

XSLT File 2: Display Activities Along with Their Participants

Purpose: This stylesheet displays all activities and lists the participants involved in each activity. The output is formatted as an HTML structure.

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

    <!-- Root template matches the entire XML document -->
    <xsl:template match="/">
        <html>
            <body>
                <h2>Activities and Participants</h2>
                <!-- Iterate over each Activity element -->
                <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>
                       <!-- Iterate over each Person element within Participants
-->
                        <xsl:for-each select="Participants/Person">
```

```
                            <!-- Display each participant's full name and email
-->
                            <li>
                                  <xsl:value-of select="concat(FirstName, ' ',
LastName, ' (', Email, ')')"/>
                            </li>
                        </xsl:for-each>
                    </ul>
                </xsl:for-each>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

activities_and_participants.xsl

XSLT File 3: Generate a Monthly Report of Service Utilization

Purpose: This XSLT stylesheet generates a summary report of the number of services provided by each service provider during a specific month. The output is an HTML table that lists the provider's name and the total number of services they were responsible for in the given month.

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

    <!-- Parameter to hold the target month for filtering -->
    <xsl:param name="targetMonth" select="'2024-08'"/>

    <!-- Define a key to group services by provider name -->
    <xsl:key name="services-by-provider" match="Service" use="Provider/Name"/>

    <!-- Root template matches the entire XML document -->
    <xsl:template match="/">
        <html>
            <body>
                        <h2>Service Utilization Report for <xsl:value-of
select="$targetMonth"/></h2>
                <!-- Begin HTML table -->
                <table border="1">
                    <tr>
                        <th>Provider</th>
                        <th>Number of Services</th>
                    </tr>
```

```
                    <!-- Iterate over services matching the target month and
group by provider -->
                    <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', $provider)[1])">
                            <tr>
                                <!-- Display the provider's name -->
                                <td><xsl:value-of select="$provider"/></td>
                                 <!-- Count and display the number of services
for the provider -->
                                 <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>
</xsl:stylesheet>
```

monthly_service_report.xsl

XSLT File 4: Display the Full Contact Information of All Providers

Purpose: This XSLT stylesheet is used to list all service providers with their complete contact information, including phone number, email, and address.

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

    <!-- Root template matches the entire XML document -->
    <xsl:template match="/">
        <html>
            <body>
                <h2>Service Providers Contact Information</h2>
                <!-- Begin HTML table -->
                <table border="1">
                    <tr>
                        <th>Provider Name</th>
```

```
                        <th>Phone</th>
                        <th>Email</th>
                        <th>Address</th>
                    </tr>
                    <!-- Iterate over each Provider element within Services -->
                                                                <xsl:for-each
select="CarePlatform/Services/Service/Provider">
                        <tr>
                            <!-- Display the provider's name -->
                            <td><xsl:value-of select="Name"/></td>
                            <!-- Display the provider's phone number -->
                            <td><xsl:value-of select="Contact/Phone"/></td>
                            <!-- Display the provider's email address -->
                            <td><xsl:value-of select="Contact/Email"/></td>
                            <!-- Display the provider's address -->
                            <td>
                                <xsl:value-of select="Contact/Address/Street"/>
<br/>
                                <xsl:value-of select="Contact/Address/City"/>,
                                <xsl:value-of select="Contact/Address/State"/>
                                                                <xsl:value-of
select="Contact/Address/PostalCode"/><br/>
                                <xsl:value-of select="Contact/Address/Country"/>
                            </td>
                        </tr>
                    </xsl:for-each>
                </table>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

provider_contact_info.xsl

XSLT File 5: Display Upcoming Activities for a Specific Participant

Purpose: This stylesheet displays all upcoming activities for a specific participant, identified by their ID. The output is formatted as an HTML list.

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

    <!-- Parameter to hold the participant ID for filtering -->
```

```
    <xsl:param name="participantID" select="'00000001'"/>

    <!-- Root template matches the entire XML document -->
    <xsl:template match="/">
        <html>
            <body>
                    <h2>Upcoming Activities for Participant ID: <xsl:value-of
select="$participantID"/></h2>
                <!-- Iterate over Activity elements that include the specified
participant -->
                                                                    <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>
                    <p><strong>Scheduled:</strong>
                          <xsl:value-of select="concat(Schedule/StartDate, ' ',
Schedule/StartTime)"/> to
                            <xsl:value-of select="concat(Schedule/EndDate, ' ',
Schedule/EndTime)"/>
                    </p>
                </xsl:for-each>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

participant_activities.xsl

XSLT File 6: Conversion to Simplified XML Format

Purpose: This XSLT stylesheet converts the original XML data into a simplified XML format required by an external system. Only essential details such as Service ID, Service Name, and Provider Name are included.

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

    <!-- Root template matches the entire XML document -->
    <xsl:template match="/">
        <SimplifiedServices>
            <!-- Iterate over each Service element -->
```

```
            <xsl:for-each select="CarePlatform/Services/Service">
                <Service>
                    <!-- Extract and include the ServiceID -->
                    <ServiceID>
                        <xsl:value-of select="ServiceID"/>
                    </ServiceID>
                    <!-- Extract and include the ServiceName -->
                    <ServiceName>
                        <xsl:value-of select="ServiceName"/>
                    </ServiceName>
                    <!-- Extract and include the ProviderName -->
                    <ProviderName>
                        <xsl:value-of select="Provider/Name"/>
                    </ProviderName>
                </Service>
            </xsl:for-each>
        </SimplifiedServices>
    </xsl:template>
</xsl:stylesheet>
```

simplified_xml_conversion.xsl

XSLT File 7: Conversion to JSON Format

Purpose: This stylesheet converts XML data related to activities, including activity ID, name, description, and participants, into JSON format for integration with a web application.

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

    <!-- Root template matches the entire XML document -->
    <xsl:template match="/">
        {
            "Activities": [
                <!-- Iterate over each Activity element -->
                <xsl:for-each select="CarePlatform/Activities/Activity">
                    {
                        <!-- Convert ActivityID to JSON format -->
                        "ActivityID": "<xsl:value-of select='ActivityID'/>",
                        <!-- Convert ActivityName to JSON format -->
                       "ActivityName": "<xsl:value-of select='ActivityName'/>",
                        <!-- Convert Description to JSON format -->
```

```
                        "Description": "<xsl:value-of select='Description'/>",
                        "Participants": [
                                <!-- Iterate over each Person element within
Participants -->
                            <xsl:for-each select="Participants/Person">
                                {
                                    <!-- Convert ID, FirstName, LastName, and
Email to JSON format -->
                                    "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>
```

xml_to_json_conversion.xsl

Appendix D: JSON Schema.

This appendix includes the JSON Schema developed to validate the structure of the JSON data generated from the XML transformations. The JSON Schema ensures that the JSON data adheres to the required format, particularly for activities and participants within the collaborative platform.

JSON Schema for Activities and Participants

```
{
    "$schema": "http://json-schema.org/draft-07/schema#",
    "type": "object",
    "properties": {
      "Activities": {
        "type": "array",
        "items": {
          "type": "object",
          "properties": {
            "ActivityID": {
              "type": "string",
              "pattern": "^[0-9]{8}$"
```

```
            },
            "ActivityName": {
              "type": "string"
            },
            "Description": {
              "type": "string"
            },
            "Participants": {
              "type": "array",
              "items": {
                "type": "object",
                "properties": {
                  "ID": {
                    "type": "string",
                    "pattern": "^[0-9]{8}$"
                  },
                  "FirstName": {
                    "type": "string"
                  },
                  "LastName": {
                    "type": "string"
                  },
                  "Email": {
                    "type": "string",
                    "format": "email"
                  }
                },
                "required": ["ID", "FirstName", "LastName"]
              }
            }
          },
                "required": ["ActivityID", "ActivityName", "Description",
"Participants"]
        }
      }
    },
    "required": ["Activities"]
  }
```

activities_participants_schema.json

Explanation

- **$schema**: Specifies the JSON Schema version being used (draft-07 in this case).
- **type**: Indicates the type of data (e.g., "object", "array", "string").

- **properties**: Defines the properties of the objects, including their expected types and patterns.
- **Activities**: An array that contains a list of activity objects. Each activity must include an ActivityID, ActivityName, Description, and a list of Participants.
- **ActivityID** and **ID**: These are strings expected to match a specific pattern (eight digits).
- **FirstName** and **LastName**: String properties representing the participant's name.
- **Email**: A string property that must conform to the standard email format.
- **required**: Lists the mandatory fields within each object.

Usage

This JSON Schema is used to validate JSON data generated from XML transformations or other sources. It ensures that the data structure meets the platform's requirements, particularly when integrating with external systems or APIs that consume JSON data.

Example JSON Data

Here's how an example JSON data structure would be validated against the provided JSON Schema:

```
{
  "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"
        }
      ]
```

```
    }
  ]
}
```

json_schema_explanation.json

This JSON data follows the structure defined by the JSON Schema, ensuring consistency and reliability when used within the platform or transmitted to external systems.

Appendix E: Python Source Code

This appendix includes the Python source code developed for the project. The code is used to process XML data, particularly for filtering services scheduled on a specific date. The provided script leverages the lxml library to parse XML, apply XPath queries, and generate human-readable output.

Python Script: Display Services Scheduled on a Specific Date

Purpose: This script filters services from the XML data based on a user-specified date and outputs the relevant details such as service name, provider, and schedule.

```
from lxml import etree

# Sample XML Data (typically, you would load this from a file)
xml_data = """
<CarePlatform>
    <Services>
        <Service>
            <ServiceID>00000001</ServiceID>
            <ServiceName>Home Care</ServiceName>
                        <Description>Personal assistance and home care
services.</Description>
            <Provider>
                <ProviderID>00000001</ProviderID>
                <Name>Caregivers Inc.</Name>
                <Contact>
                    <Phone>+1112223333</Phone>
                    <Email>contact@caregivers.com</Email>
                    <Address>
                        <Street>789 Elm St</Street>
                        <City>Metropolis</City>
                        <State>Regionburg</State>
                        <PostalCode>54321</PostalCode>
```

```
                        <Country>Landville</Country>
                    </Address>
                </Contact>
            </Provider>
            <Schedule>
                <StartDate>2024-08-01</StartDate>
                <EndDate>2024-12-31</EndDate>
                <StartTime>08:00:00</StartTime>
                <EndTime>18:00:00</EndTime>
            </Schedule>
        </Service>
        <Service>
            <ServiceID>00000002</ServiceID>
            <ServiceName>Medical Transport</ServiceName>
                        <Description>Transportation   services   for   medical
appointments.</Description>
            <Provider>
                <ProviderID>00000002</ProviderID>
                <Name>HealthTrans Co.</Name>
                <Contact>
                    <Phone>+4445556666</Phone>
                    <Email>support@healthtrans.com</Email>
                    <Address>
                        <Street>101 Pine St</Street>
                        <City>Capitol City</City>
                        <State>Stateville</State>
                        <PostalCode>13579</PostalCode>
                        <Country>Countryland</Country>
                    </Address>
                </Contact>
            </Provider>
            <Schedule>
                <StartDate>2024-08-10</StartDate>
                <EndDate>2024-12-31</EndDate>
                <StartTime>07:00:00</StartTime>
                <EndTime>20:00:00</EndTime>
            </Schedule>
        </Service>
    </Services>
</CarePlatform>
"""

# Parse the XML data
root = etree.fromstring(xml_data)
```

```python
# Define the target date
target_date = "2024-08-10"

# Find services scheduled on the target date
services = root.xpath(f"//Service[Schedule/StartDate = '{target_date}']")

# Print out the results
print(f"Services scheduled on {target_date}:\n")

for service in services:
    service_name = service.find("ServiceName").text
    provider_name = service.find("Provider/Name").text
    start_time = service.find("Schedule/StartTime").text
    end_time = service.find("Schedule/EndTime").text

    print(f"Service Name: {service_name}")
    print(f"Provider: {provider_name}")
    print(f"Start Time: {start_time}")
    print(f"End Time: {end_time}\n")
```

filter_services_by_date.py

Explanation

- **lxml Library**: The lxml library is a powerful tool for processing XML and HTML in Python. It is used here to parse the XML data and apply XPath queries to filter and retrieve specific information.
- **XML Parsing**: The XML data is parsed into an element tree using etree.fromstring(). This tree structure allows for easy navigation and querying of the XML elements.
- **XPath Querying**: The script uses an XPath query to filter services based on the StartDate element. The target_date is specified as a variable, making the script adaptable to different dates.
- **Output**: The script outputs the relevant service details for the specified date, including the service name, provider, start time, and end time.

Usage

To run this script:

1. **Install the lxml Library**: If you haven't already installed lxml, you can do so using pip:

```
PS C:\Users\benab\Downloads\Ismail Ben Abdelkader\DSTI\Data Pipeline> pip install lxml
Requirement already satisfied: lxml in c:\users\benab\downloads\ismail ben abdelkader\dsti\data pipeline\project\.venv\lib\site-packages (5.3.0)
```

2. **Run the Script**: Save the script to a .py file and run it using Python. The output will display the services scheduled on the specified date.

Example Output

Given the sample XML data and the target date 2024-08-10, the script would output:

```
PS C:\Users\benab\Downloads\Ismail Ben Abdelkader\DSTI\Data Pipeline\Project\Appendix\E> python filter_services_by_date.py
Services scheduled on 2024-08-10:

Service Name: Medical Transport
Provider: HealthTrans Co.
Start Time: 07:00:00
End Time: 20:00:00
```

This output provides a clear and concise view of the services scheduled for the selected date.

Appendix F: XSLT Transformation Outputs.

This appendix showcases the outputs generated by the XSLT transformations developed for the project. Each output corresponds to a specific scenario and demonstrates how the XML data was transformed into various formats, such as HTML and JSON.

Output 1: Display All Services and Their Providers

Description: This output is an HTML table that lists all services offered on the platform, along with their respective providers. The table includes columns for the service name, description, provider name, and contact email.

HTML Output

```
<html>
    <body>
        <h2>List of Services and Providers</h2>
        <table border="1">
            <tr>
                <th>Service Name</th>
                <th>Description</th>
                <th>Provider Name</th>
                <th>Contact Email</th>
            </tr>
            <tr>
                <td>Home Care</td>
                <td>Personal assistance and home care services.</td>
                <td>Caregivers Inc.</td>
                <td>contact@caregivers.com</td>
```

```
            </tr>
            <tr>
                <td>Medical Transport</td>
                <td>Transportation services for medical appointments.</td>
                <td>HealthTrans Co.</td>
                <td>support@healthtrans.com</td>
            </tr>
        </table>
    </body>
</html>
```

services_providers_output.html

Output 2: Display Activities Along with Their Participants

Description: This output is an HTML structure that lists all activities organized on the platform, along with the participants involved in each activity. Each activity is followed by a list of participants with their names and email addresses.

HTML Output

```
<html>
    <body>
        <h2>Activities and Participants</h2>
        <h3>Weekly Social Gathering</h3>
         <p>A weekly social event for elderly people to interact and engage in various activities.</p>
        <h4>Participants:</h4>
        <ul>
            <li>John Doe (john.doe@example.com)</li>
            <li>Jane Smith (jane.smith@example.com)</li>
        </ul>
    </body>
</html>
```

activities_participants_output.html

Output 3: Display Services Scheduled on a Specific Date

Description: This output is an HTML table that lists services scheduled on a specific date, including details such as the service name, provider, start time, and end time.

HTML Output

```
<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>Medical Transport</td>
                <td>HealthTrans Co.</td>
                <td>07:00:00</td>
                <td>20:00:00</td>
            </tr>
        </table>
    </body>
</html>
```

services_scheduled_output.html

Output 4: Display the Full Contact Information of All Providers

Description: This output is an HTML table that lists the full contact information of all service providers, including their phone numbers, email addresses, and physical addresses.

HTML Output

```
<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>
```

providers_contact_info_output.html

Output 5: Display Upcoming Activities for a Specific Participant

Description: This output is an HTML structure that lists all upcoming activities for a specific participant, identified by their ID. The list includes the activity name, description, and schedule.

HTML Output

```
<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>
            <p><strong>Scheduled:</strong>  2024-08-10  10:00:00  to  2024-12-31
14:00:00</p>
    </body>
</html>
```

participant_activities_output.html

Output 6: Conversion to Simplified XML Format

Description: This output is a simplified XML format that includes only the essential details for services, such as the Service ID, Service Name, and Provider Name. This format is designed for integration with an external system.

Simplified XML Output

```
<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>
```

simplified_services_output.xml

Output 7: Conversion to JSON Format

Description: This output is a JSON structure generated from the XML data, specifically for activities and participants. The JSON format is useful for integration with web applications that require data in JSON.

JSON Output

```
{
    "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"
                }
            ]
        }
    ]
}
```

activities_json_output.json

Explanation

Each output provided in this appendix corresponds to a specific XSLT transformation scenario detailed earlier in the report. The outputs demonstrate how the original XML data was transformed into different formats, such as HTML and JSON, to meet the project's requirements for data visualization and integration.