The main class creates a TrustManager that uses a Cert read in from the resources folder instead of being installed in the JVM's cacerts.

Used the TrustManager in two calls:

1. To the OAuth2 URI to get the token.

2. To the URI that has the data.

****

1. This code makes a call to get a new OAuth2 token every time it's called (tokens are good for an hour).  
 Should store the token somewhere for reuse and only make a call to OAuth2 when it has expired.

2. The synchronous HttpClient send is being used. Need to explore if using HttpClient's asynchronous send would be a performance improvement.

****

Initially used a sample XML to create a schema, then used JAXB to create sub-classes for unmarshalling.  
JAXB created 6 domain classes and an ObjectFactory (did not use).

****  
  
Before creating the trustAllCerts() method – which reads in a Cert from the (canonical?) /resources folder – I trusted all Certs by creating the following TrustManager:

// Trusts Certs - No SSL validation

TrustManager[] trustAllCerts =

new TrustManager[] {

new X509TrustManager() {

@Override

public void checkClientTrusted(X509Certificate[] chain, String authType)

throws CertificateException { /\* Empty method \*/ }

@Override

public void checkServerTrusted(X509Certificate[] chain, String authType)

throws CertificateException { /\* Empty method \*/ }

@Override

public X509Certificate[] getAcceptedIssuers() { return new X509Certificate[0]);

} // new X509TrustManager()

}; // TrustManager[]

****

The JAXB unmarshal class:

import jakarta.xml.bind.JAXBContext;

import jakarta.xml.bind.JAXBException;

import lombok.extern.slf4j.Slf4j;

import pal.schemamap.EmployeeList;

import java.io.StringReader;

@Slf4j

public class PalXMLToObject {

public EmployeeList unmarshall(final String palXml) {

var employeeList = new EmployeeList();

try {

var jaxbContext = JAXBContext.*newInstance*(EmployeeList.class);

var jaxUnmarshaller = jaxbContext.createUnmarshaller();

employeeList = (EmployeeList)

jaxUnmarshaller.unmarshal(new StringReader(palXml));

}

catch (JAXBException e) {

*log*.error("Error in unmarshalXML is {}", e.toString());

}

return employeeList;

}

}