CS 452 – Introduction to the Science and Technology of Services

Winter semester 2022

***Responsible instructor:** Pitikakis Marios (pitikakis@csd.uoc.gr)*

2nd Assignment
Due date: 20/11/2022

General instructions

Write a report demonstrating your results using images, screenshots, code and any other explanation necessary. Include a cover page with your student ID and name. The filename of the deliverable must be in the following form: **ask2_AM** (where AM is your student id number e.g. ask2_1234). Any images/screenshots should be included in your report document. Do NOT submit separate image files or code. The report document must be in pdf format. Submit a single compressed file (zip/rar/gz etc) if your work contains multiple files. Note that this assignment is to be done **individually**. Cheating in any way, including giving your work to someone else, will result in failing this assignment (a mark of zero will be given).
Submit your assignment **electronically** (online) by 20/11/2022 at 23:59 using the course webpage. You should also be prepared to present your work in case an oral examination is scheduled shortly after the submission date.

Overview

In this assignment, you will work with XML and create/deploy mainly REST-style web services. For the REST-style web services, you will use the Jersey framework and Spring Boot. As a bonus exercise, you will experiment with SOAP-based web services. For the SOAP/WSDL Web Services, you will use the Apache Axis2 service framework in Eclipse using Apache Tomcat as an application server.

Background

We strongly suggest you read/review the following resources:

- Course slides
- The lab tutorials on Rest-style and SOAP-based Web Services
- Spring Getting Started Guides
- Eclipse Web Tools Platform User Guide
- Jersey User Guide
- Swagger

Additional help or tutorials can be found online.

Submit only the java files you created for the web services and the clients. In your assignment report, you should include any implementation details worth mentioning and screenshots of requests/responses when invoking these services.

Tasks

A. (**15 points**) In the *email* example shown below, we specify the body of an email to containing exactly one text and a number of attachments. Modify this XML schema to allow for an arbitrary number of texts and attachments in any order.

```
<element name="email" type="emailType"/>

<complexType name="emailType">
       <sequence>
              <element name="head" type="headType"/>
              <element name="body" type="bodyType"/>
       </sequence>
</complexType>
<complexType name="headType">
       <sequence>
              <element name="from" type="nameAddress"/>
              <element name="to" type="nameAddress" minOccurs="1" maxOccurs="unbounded"/>
              <element name="cc" type="nameAddress" minOccurs="0" maxOccurs="unbounded"/>
              <element name="subject" type="string"/>
       </sequence>
</complexType>
<complexType name="nameAddress">
       <attribute name="name" type="string" use="optional"/>
       <attribute name="address" type="string" use="required"/>
</complexType>
<complexType name="bodyType">
       <sequence>
              <element name="text" type="string"/>
              <element name="attachment" minOccurs="0" maxOccurs="unbounded">
              <complexType>
                     <attribute name="encoding" use="default" value="mime">
                            <simpleType>
                                   <restriction base="string">
                                          <enumeration value="mime"/>
                                          <enumeration value="binhex"/>
                                   </restriction>
                            </simpleType>
                     </attribute>
                     <attribute name="file" type="string" use="required"/>
              </complexType>
              </element>
       </sequence>
</complexType>
```

B. (**35 points**) Using the Jersey framework, do the following:

i) Create a REST-style web service for converting cm to inch and vice versa. It will contain two operations called inch_to_cm and cm_to_inch.

ii) Create a service client as a Java application to invoke OpenWeatherMap services. Study the OpenWeatherMap API guide (you will need to make a free account and get a key). More specifically, your client should be able to get weather information for a specific city (e.g. Heraklion) and return the results in XML format.

iii) [optional/bonus] Create a service client as in (**ii**) that can handle the JSON response from the OpenWeatherMap API and displays only the city name and the corresponding humidity information (JSON processing).

C. (**50 points**) Using the Spring Boot framework, do the following:

i) Create a REST-style web service for currency conversion and currency rates. For currency conversion, the endpoint will be called "currencyconvert" and the required input parameters are "from", "to" and "amount". The "from" and "to" parameters take as input a three-letter currency code (e.g. EUR, GBP, USD etc). Use a static currency conversion rate (e.g. today's rates) and at least five different currencies. The response objects will contain all the input information and an object called "result" (i.e. the conversion result). For currency rates, the endpoint will be called "currencyrates" and the required input parameters are "from" and "to". The response objects will contain all the input information and an object called "rate" (i.e. the conversion rate).

ii) Include the API documentation of your web service using Swagger (using the OpenAPI 3 specification).

iii) Deploy your web service on the AWS platform.

iv) Create a service client as a Java application that consumes your deployed (AWS) REST web service and displays the information on the console in a human-readable format (or generates a web page).

Bonus Task

C. (**30 points**) Using the Apache Axis2 service framework (as a SOAP / WSDL engine)
i) Create two web services:
(ws1) a service converting cm to inch and vice versa and

(ws2) a service calculating the ideal weight (in kg) based on height (in cm).

These services will be deployed on the same server.

More specifically, (ws1) will contain two operations called inch_to_cm and cm_to_inch. (ws2) will contain one operation called idealWeightFormula, which will take as input the height in cm and will calculate the ideal weight for men and women as follows (Miller formula):

Men: Ideal Body Weight (kg) = 56.2 kg + 1.41 kg **per every inch over 5 feet**
Women: Ideal Body Weight (kg) = 53.1 kg + 1.36 kg **per every inch over 5 feet**

The response messages of (ws2) will include two parameters: men's and women's weight. Example response for height 175 cm :

```
<ns:millerFormulaResponse xmlns:ns="http://intro.ws.hy452">
	<ns:return xmlns:ax21="http://intro.ws.hy452/xsd" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xsi:type="ax21:IdealWeight">
		<ax21:menWeight>68.745475</ax21:menWeight>
		<ax21:womenWeight>65.2006</ax21:womenWeight>
	</ns:return>
</ns:millerFormulaResponse>
```

ii) Create two service clients as Java applications to demonstrate the invocation and the responses of the developed (ws1) and (ws2) web services.