CMPE 273 – Enterprise Distributed Systems

Lab 3 – Web service

Due Date: 6 November 2016

This lab assignment covers developing SOAP Web services using. This lab assignment is graded based on 30 points and is an **individual effort** (e.g.: no teamwork allowed)

Prerequisites

- You must have carefully read the Environment Setup document. You should be able to run the "Hello World" web service described in Environment Setup document.
- You must know Java core, collection API and JDBC.

The Assignment

You will be developing two clients and servers during this lab.
On the due date, turn in the following (via canvas):

- A code listing of each of your clients/servers
- Screen captures of each client/server during execution

Grading

- Server1/Client1 (10pts) and Server2/Client 2 (20pts)
- Late assignments will be accepted, but will be subject to a penalty of -5 points per day late:
- Submissions received at or before the class on the due date can receive up to 30 pts maximum.
- Submissions shall include source code for each client/server pair, plus screen image captures of each client/server pair during and after run.

Server 1 - "Calculator" to demonstrate web services

The first Web service based server you need to develop is the "Calculator". This server should perform the following tasks:
1. Addition
2. Subtraction
3. Multiplication
4. Division

Take care of exceptions.

Client 1 – "Calculator Client"
Creation of Client is completely independent of any sub element of Java technology. Students can make their client using HTML5 ,Angular.Js.

The client should provide the facility to test all the functions of calculator service.

The client for Server 1 should behave in the following way: Perform each operation of Server once. Print out the result in a systematic format, then automate the following process and print the average times as mentioned below.

1. Start a timer
2. Invoke 1,000 math solver calls on randomly selected tasks.
3. Stop the timer and print out the average time to perform each operation

1. Start a timer
2. Invoke 5,000 math solver calls on randomly selected tasks.
3. Stop the timer and print out the average time to perform each operation

1. Start a timer
2. Invoke 100 concurrent users with 1000 calls each to calculator on randomly selected tasks.
3. Stop the timer and print out the average time to perform each operation

Your output should be the average time for each operation.

Perform analysis for the average time among three cases (if you see latency increase or decrease more than 10%, then explain what happened and prove your statement) and also compare the results with REST from lab1

Part 2: Ebay Marketplace Application (20 Points)

You need to develop "Simple Market Place". This server should perform the following tasks:

a) Basic functionality would include user to Sign Up, Sign In, Sign Out. Sign Up should have first name, last name, Email and password. In order to use the system, a user must sign in first.
b) Should be able to store user's advertisements for others to read. This should at-least include the item name, item description, seller information, item price and quantity. (Only text, no images).
   E.g. Advertisement – "Laptop", "2.2 GHz Core 2 Duo, 2GB RAM…", "Jon Smith, shipping from NY", "$600", "4 pieces"
c) Should give all the advertisements details to all the other users.
d) Users can buy the product displayed by other users. You should take care of quantities and should respectively reflect user accounts in case of any transaction.
e) Shopping Cart should be maintained which will reflect temporary items. Users should be able to add, remove items from the cart until checkout.
f) Checkout should deduct and add items from seller and buyer respectively.
g) Should perform simple credit card validations on payment. (check to be sure the number entered is16 digits)
h) Users account should reflect all the bought and sold items.
i) Should maintain time last logged in and should be returned back when user logs in.

The Service should take care of exception that means validation is extremely important for this server. Good Exception Handling and validation would attract good marks.

Client: "Simple Market Place Client"

In this part you have to build a prototype "Ebay Application" using Node.js, HTML5 and AngularJS.

Client must include all the functionalities implemented by the web services.
Client similar to eBay will attract good marks.

Server: Ebay to demonstrate stateful web services **(12 Points)**
Server should perform following tasks:

1. **Basic User functionalities:**
    a. Sign up the new users (First name, Last name, Email, Password). Passwords have to be encrypted
    b. Sign in with existing users
    c. Sign out
2. **Profile:**
    a. About: Birthday, Ebay handle, contact information and location
    b. Should be able to store user's advertisements for others to read. This should at-least includes the item name, item description, seller information, item price and quantity. (Only text, no images).

    E.g. Advertisement – "Laptop", "2.2 GHz Core 2 Duo, 2GB RAM…", "Jon Smith, shipping from NY", "$600", "4 pieces"

    c. Should give all the advertisements details to all the other users.
    d. Users can bid the product displayed by other users. Bidding takes place over 4 days. For some items, you can purchase without bidding. You should take care of quantities and should respectively reflect user accounts in case of any transaction.
    e. Shopping Cart should be maintained which will reflect temporary items. Users should be able to add, remove items from the cart until checkout.
    f. Checkout should deduct and add items from seller and buyer respectively.
    g. Should perform simple credit card validations on payment. (check to be sure the number entered is16 digits)
    h. Users account should reflect all the bought and sold items.
    i. Should maintain time last logged in and should be returned back when user logs in.
    j. User tracking: Generate logs into a file when user clicks any place in the web page. There are generally two major logs. Event Logs: record timestamp, userid, click object id, any descriptions. Bidding Logs (periodic logs of bidding process) timestamp, item id, user id, bid amount.

3. Implement Connection pooling for database access

Validation is extremely important; exception handling should be implemented. **Implementations with proper validations and exception handling will attract good marks.**

Client: Ebay Client **(5 Points)**
Client must include all the functionalities implemented by the web services. Develop the Client using HTML5 and AngularJS. **Client similar to Ebay will attract good marks.**]

Testing of the server should be done using JMeter and Mocha. Mocha is a node.js testing framework.

Following tasks to be tested using JMeter: **(2 Points)**

Test the server for **100, 200, 300, 400 and 500 concurrent users** with and without connection pooling. **Draw the graph with the average time and include it in the report.**

Following tasks to be tested using Mocha: **(1 Point)**

Implement 5 randomly selected SOAP web service API calls using Mocha. **Display the output in the report.**

Create private repository on the GitHub or bitbucket to manage source code for the project. Add description for every commit. Description should consist of one-line overview on what is committed. Include GitHub/bitbucket project link with access credentials in your report.

Deliverables Required:

- Submission should include only source code (no node modules or lib should be included)

- Project directory should include the group ID/Name (e.g.: Lab1-caffiene)
- Archive the report and source in one archive file (e.g.: ZIP file)
- Do not submit binaries, node modules, .class files or supporting libraries **(3 Points will be deducted if included)**
- Project Report:
    - Introduction: Goals and purpose of your system
    - System Design: Describe your chosen design
    - Results: Screen captures of Part1 and Part2 Client/Servers
    - Performance graphs. Analyze the graphs and explain the results
    - Answers to Questions in Part3
- Sample directory structure for submission (for person with name Smith) Archive the below files in ZIP file with name: **Lab1-Smith.zip**
    - **Smith-lab1-report.doc**
    - **Lab1/ (directory, do not include libs)**
- **Online Submission only –** Submissions should be made on Canvas before the due date.