Assignment 6: JavaScript and Cloud Computing

Release Date: November 27, 2017
First Deadline: December 8, 2017
Due Date: December 10, 2017 (11:59 pm Central Time)
(This is an individual assignment)

Objective:
In this assignment you will learn about JavaScript and Cloud Computing.

High-level Description:
Using HTML, JavaScript, and REST APIs build the following "analytics" display for *Solum's meetings*
The output should be as follows:

Year	Number of meetings
2013	<number of meetings in 2013>
2014	<number of meetings in 2014>
2015	<number of meetings in 2015>
2016	<number of meetings in 2016>
2017	<number of meetings in 2017>

3) Details:

Part I: Calculating meeting counts
Solum team meeting logs are available at following *source:*
http://eavesdrop.openstack.org/meetings/solum_team_meeting/
For each meeting four different files are generated. For example, if you check this link:
http://eavesdrop.openstack.org/meetings/solum_team_meeting/2013/
You will notice there are four files (.log.html, .log.txt, .html, .txt) corresponding to a meeting date. For instance, for 2013-11-12, there are following files in the folder:
solum_team_meeting.2013-11-12-16.00.html
solum_team_meeting.2013-11-12-16.00.log.html
solum_team_meeting.2013-11-12-16.00.log.txt
solum_team_meeting.2013-11-12-16.00.txt
For calculating the counts, you can either use the meeting names, or the last modified timestamp. But you need to ensure that you count each meeting date only once in the overall meeting count. So essentially, the meeting on 2013-11-12 should contribute 1 to the overall meeting count, even though there are four files generated for each meeting.

Part II: HTML, JavaScript, REST API
The entry point to your web application should be a HTML file whose URL should be:
http://localhost:8080/openstack-meeting-analytics.html

This HTML page should have following text (you don't have to worry about making the text color blue):

Welcome to OpenStack Solum Meeting Analytics
Click here to see number of meetings for Solum OpenStack project

Clicking on “here” should show the analytics display table.
You are required to use HTML and JavaScript for building the UI (the initial HTML page and the page that displays table).
You will need to build a REST API that can be invoked from the JavaScript to get the meeting count data, which will be rendered by the JavaScript.

Part III

Deploy your web application in the “cloud” and make it accessible over the Internet. You can use any of the several commercial cloud services for this, such as:

- Amazon AWS (http://aws.amazon.com/)
- Heroku (https://www.heroku.com/)
- Your-choice-of-cloud-provider

Once deployed, your web application should be available at:
http://<IP-address-of-your-VM>:8080/openstack-meeting-analytics.html or
http://<platform-specific-url>/openstack-meeting-analytics.html

4) Design details:
You will have to expose a suitable REST API that can be used by your web application’s JavaScript AJAX requests. It is up to you how you define a REST API for this purpose. Similarly, it is up to you how you calculate meeting counts.

Your web application should be publicly accessible. If it is not, then for grading we will consider if the app runs correctly on a non-public setup. But you will not get points for cloud deployment (point split to be decided).

Submission Details:
1) Create the folder structure as follows:
assignment6/src/main/java/<Java classes>
assignment6/src/main/webapp/<*.html>
assignment6/src/main/webapp/<*.js>
assignment6/src/main/webapp/WEB-INF/web.xml
assignment6/pom.xml
assignment6/README.txt

Feel free to add more packages inside the “src/main/java” to segregate work into different layers.
Also, if above folder structure is not working for you and having WebContent folder as previous assignments is working, then use that.

2) Use following format for README.txt
name: <your name>
eid: <your ut eid>
bitbucketid: <your bitbucket id>
appURL: <Public URL of your application>
cloud: <Name of the cloud provider you chose>
comments: <Comments, if any>

Create a *private* repository on bitbucket (https://bitbucket.org/product/pricing?tab=host-in-the-cloud sign up using the free option if you don't have bitbucket account). Name the repository assignment6 and push/submit all the files that you create for your assignment to this repository.

Grant “read” access to me and Sailesh
- Usernames: devdattakulkarni, svsailesh

We will use the latest commit ID for grading. You don't need to submit anything on canvas.

Helpful Material:
Lecture Notes: JavaScript, Cloud computing, Webapp-deployments-to-public-clouds

Relevant projects from github repository (https://github.com/devdattakulkarni/ModernWebApps/):
JavaScript-Example

Video showing how to deploy web app on cloud VM: https://vimeo.com/126759241

Deadlines:
First deadline: Initial submission which is will be graded for 3 points. You don't need to have assignment implementation completed by this date. This means if your first commit is on or before December 8 11:59 pm, you will receive points out of 100. If your first commit is after that then you will receive points out of 97. This deadline is to ensure that you complete all the required setup (JavaScript sample example setup, Eclipse, Tomcat, etc.) early and not postpone it until last minute.

Due date: Deadline by which your assignment needs to be completed (December 10, 11:59pm)

Late penalty:
5 points for each late day after the due date.

Collaboration policy:
This is an individual assignment. You are allowed to discuss concepts and high-level implementation questions with each other. But you are not allowed to copy or share code with each other or students who might have taken this class before. Final submission should be your own code.