Lab 5 Specification – Exploring Web Application Development
Due (via your git repo) no later than 2 p.m., Monday, 19th Oct 2020.
50 points

Lab Goals

- Refresh our knowledge in using OpenStack
- Implement web application (services) on the Cloud Platform.

Summary

We will do a few hands-on exercises to use a virtual machine in the Cloud Platform, and implement services on the new instance. We had started discussing the techniques, by which we can publish Cloud services. It is the right time to understand how the web application setup works practically, and doing the basic networking to achieve this goal. Additionally, we will watch a video clip, which is a segment of a ted talk on Web Applications, to further solidify our understanding.

Learning Assignment

If not done so already, please read all of the relevant ”GitHub Guides”, available at https://guides.github.com/, which explains how to use many of the features that GitHub provides. In particular, please make sure that you have read guides such as ”Mastering Markdown” and ”Documenting Your Projects on GitHub”; each of them will help you to understand how to use both GitHub and GitHub Classroom. To do well on this assignment, you should also read

- **Servlets reading material in the class (lesson-4 and part-3) repository.**
- **If not done already, Basic Networking reading material in the class repository.**

Assignment Details

Now that we have discussed some fundamental principles behind web application and publishign Cloud Services, it is now time to implement some challenging requirements.

It is required for all students to follow the honor code. Some important points from the class honor code are outlined below for your reference:

1. Students are not allowed to share code files and/or other implementation details. It is acceptable to have a healthy discussion with your peers. However, this discussion should be limited to sharing ideas only.
2. Submitting a copy of the other’s program(s) is strictly not allowed. Please note that all work done during laboratory sessions will be an opportunity for students to learn, practice, and master the materials taught in this course. By doing the work individually, students maximize the learning and increase the chances to do well in other assessments such as lab assignments, skill tests, projects, etc.

At any duration during and/or after the lab session, students are recommended to team up with the Professor and/or the Technical Leader(s) to clarify if there is any confusion related to the items in the lab sheet and/or class materials.

Section 1: Cloud Computing

This section is worth 15 points. The points breakdown is provided below:

- Task 1 = 15 points, a maximum of 3 points awarded for each question.

Greg Plum has been involved in channel development since 2001 and has enjoyed building channel sales operations from scratch. He currently serves as the principal of Plum Unified Communications, a faculty member of CompTIA, and a Microsoft training specialist for Brainstorm, Inc. Plum's passion for the channel, as well as his desire to form long-term relationships with his partners, has gained him the honor of being named a Channel Executive of the Year Finalist in 2009, 2010, and 2011. Plum lives in Delaware with his wife and two children. He enjoys spending time with his family, distance running and everything about the beach! (text taken from YouTube description).

This video is a segment from the series of talks given at a TEDx event in a TED conference. In this video, Greg presents his view on Cloud Computing. In this section, we will watch a video and reflect on the points discussed in the clip. This reflection is instrumental to further advance our understanding of Cloud Computing and in general to foster the learning from our recent discussions on how to publish cloud services. To complete this part, it is required to do the following:

- **Task 1:** Watch this short video clip, by using the link below:

  `https://www.youtube.com/watch?v=KoM0xmb7ss0`

  After watching the video, create a markdown file and name it as `video-reflection`. In this file, provide detailed answers to the questions provided below:

  1. How is Cloud Technology useful to travelers?
  2. How does the speaker connect Uber to Cloud services?
  3. What is Internet Of Things?
  4. Professor Mohan, repeatedly mention this statement in our classes, "So far, we did computing inside our laptops. But, in CMPSC 402, we will learn and do computing, outside our laptop.". What did the speaker say in this video, that is similar to the point mentioned in class, about computing outside our laptop.
  5. What are the examples of the rate of change of technology mentioned in this video? And what did the speaker say that we should do to address this change.

Section 2: Web Application Development

This section is worth 35 points. The points breakdown is provided below:

- Task 2 = 15 points
- Task 3 = 20 points

In this section, we will setup web server on OpenStack and implement a web application on the virtual machine. The underlying principle that tried out here are the publishing software as a service. The web application is the service (rendered from server to the client). To complete this part, it is required to complete the tasks listed below:

1. **Task 2:**
   - Make sure that the virtual machine is in running state in OpenStack.
   - By using your OpenStack pem file, SSH to the virtual machine. Terminal may be used for mac/linux users, and git bash for windows users.
   - Make sure that Java (JDK) is installed on the virtual machine. Follow the steps from the previous labs and practicals to make sure that you can execute a HelloWorld java program on the Cloud Machine.
   - Run the following commands on your virtual machine:

     sudo apt update

     sudo apt install tomcat9
   - It is nor required to run these commands. But these commands may be used to control the tomcat service on the virtual machine.

     sudo service tomcat9 stop

     sudo service tomcat9 start

     sudo service tomcat9 status
   - At this point, Apache Tomcat is running on your virtual machine. You may type in this on your browser (client), to verify the Tomcat installation.

     `http://141.195.7.174:8080/`
   - In the URL, above, please make sure to change the IP address to your OpenStack public ip address.
   -
   - Initiate the transfer of the web application to the server. First make sure that the open stack pem file is placed in the age-app folder. Edit the move.sh file to include the correct ip address of your open stack machine, and the correct pem file name for your open stack profile. Navigate to the starter-code repository, and execute the move.sh file in terminal (mac/linux), and git bash (windows) users (Manually).
   - Once the transfer is completed, then execute the following commands to setup the web application on the virtual machine's tomcat server.

sudo mv age.war /var/lib/tomcat9/webapps

cd /var/lib/tomcat9/webapps

sudo chmod go+rx age

cd age

./compile.sh

- At this point, the servlet code will be compiled and the class files will be moved to the corresponding directory.
- If all the previous steps are correctly completed, then the following url should work. Please change the ip address to your ip address.

  `http://141.195.7.174:8080/age/index.jsp`
- Once the form is filled out, and by clicking on submit button, the servlet will be invoked and the year of birth will be displayed on the following page.
- Upload a screenshot of both pages, index.jsp and message.jsp, from the browser window, and name the file as `age-app-8080`.
- Next, we will do some networking changes on the server.
- Open the server configuration file and change the port from 8080 to 80. Type the following commands:

  sudo nano /etc/tomcat9/server.xml
- Change the port number from 8080 to 80. Scroll through the server.xml and look for Connector port="8080". Change "8080" to "80". Keep the double quotes, and change just the number.

  sudo service tomcat9 stop

  sudo service tomcat9 start
- If all the previous networking steps are correctly completed, then the following url should work. Please change the ip address to your ip address.

  `http://141.195.7.174/age/index.jsp`
- Note: The default port is 80 for http requests.
- Once the form is filled out, and by clicking on submit button, the servlet will be invoked and the year of birth will be displayed on the following page.
- Upload a screenshot of both pages, index.jsp and message.jsp, from the browser window, and name the file as `age-app-80`.
- Next, let us do implement the same application using get method.
- Open the index.jsp file, and edit line no 29, form attributes, and add a new attribute called onclick = "invoke();" method = "post" to method = "get".
- Edit line no 53, form attributes, and add a new attribute called onclick to invoke the javascript functional called invoke. This javascript function will invoke the servlet internally.

  ¡input type="submit" id="btnSubmit" value="Submit"/¿

  ¡input type="submit" id="btnSubmit" value="Submit" onclick = "invoke();"/¿
- Open the AgeServlet code in src/com/web folder. Copy the lines between 36 and 44. Comment these lines. Paste the copied lines to the doGet() method.
- At this point, create a new WAR file, move the WAR file to the server, and do the setup.
- Initiate the transfer of the web application to the server. Edit the move.sh file to include the correct ip address of your open stack machine, and the correct pem file name for your open stack profile. Navigate to the starter-code repository, and execute the move.sh file in terminal (mac/linux), and git bash (windows) users (Manually).
- Once the transfer is completed, then execute the following commands to setup the web application on the virtual machine's tomcat server.

```
sudo rm -rf /var/lib/tomcat9/webapps/age.war
sudo rm -rf /var/lib/tomcat9/webapps/age
sudo mv age.war /var/lib/tomcat9/webapps
cd /var/lib/tomcat9/webapps
sudo chmod go+rx age
cd age
./compile.sh
```

- At this point, the servlet code will be compiled and the class files will be moved to the corresponding directory.
- If all the previous steps are correctly completed, then the following url should work. Please change the ip address to your ip address.

  `http://141.195.7.174/age/index.jsp`
- Once the form is filled out, and by clicking on submit button, the servlet will be invoked and the year of birth will be displayed on the following page.
- Upload a screenshot of both pages, index.jsp and message.jsp, from the browser window, and name the file as `age-app-get`.

2. **Task 4:**

   - Make a copy of the index.jsp file and name the file as login.jsp.
   - Make the following edits to the login.jsp file:
     (a) Modify the **form** method from get to post.
     (b) Introduce changes to the design, by changing name fields to username, and age fields to password. Make sure to change all the references throughout the login page, including labels, input textboxes, and javascript invoke function.
   - Make a copy of message.jsp and name the file as login-status.jsp.
   - Create a new file called LoginServlet, using a similar format as AgeServlet. Do the following change:
     (a) In the doPost() method, setup the validations to check if the username is ”402-user”, and password is ”402-pass”.
     (b) If the validation is success, populate the message as ”Login success”.
     (c) If the validation failed, then populate the message as ”Login failed”.
     (d) Edit the file to forward request to ”login-status.jsp”, instead of ”message.jsp”.
   - Make the required changes to the WEB-INF/web.xml file:
     (a) Introduce a new servlet and servlet mapping tags. You may create a duplicate of AgeHandler.
     (b) Edit the new servlet and servlet mapping tags, by changing Age to Login respectively. For example, you will have LoginHandler in addition to the AgeHandler.
   - If all the previous steps are correctly completed, then the following url should work. Please change the ip address to your ip address.

     `http://141.195.7.174/age/login.jsp`
   - Once the form is filled out, and by clicking on submit button, the servlet will be invoked and the login success or failure will be displayed on the following page.
   - Upload a screenshot of both pages, login.jsp and login-status.jsp, from the browser window, and name the file as `login-app`.

Submission Details

For this assignment, please submit the following to your GitHub lab repository.

1. **video-reflection** markdown file.
2. upload of **age-app-8080** file.
3. upload of **age-app-80** file.
4. upload of **login-app** file.
5. upload of **age-app-get** file.
6. It is recommended to upload a readme file, with the details that you would like the Professor to know while grading the work. For example, it may be reflection of your experience in the lab by highlighting some of the challenges faced and a brief mention of how you had addressed those challenges while implementing this lab. The readme file may also include a brief mention of any details that one should know about executing your program and what to expect during the execution.
7. It is highly important, for you to meet the honor code standards provided by the college and to ensure that the submission is made before the deadline. The honor code policy can be accessed through the course syllabus. Make sure to add the statement "This work is mine unless otherwise cited." in all your deliverables such as source code and PDF files.

Grading Rubric

1. Details including the points breakdown are provided in the individual sections above.
2. If a student needs any clarification on their lab credits, it is strongly recommended to talk to the Professor. The lab credits may be changed if deemed appropriate.