SWE 642: Assignment 3

Servlet, JSPs, MVC & Database I/O

This homework can be done individually or in a group of maximum 4 students. If you work in a group, please identify the contribution you made to complete the work. Also, document any obstacles you came across when implementing this homework, the resolution(s) to address the obstacles, and any lessons learned/best practices.

This assignment focuses on server-side processing of the Student Survey Form data via MVC implementation using a RequestDispatcher object. The implementation requires you to implement one Servlet that acts as a front controller that receives all client requests, saves the form data to a database table, performs business logic(s) via business delegate classes, stores JavaBean objects into a session or request object, and then forwards the request to appropriate JSP to present the data to the user. The assignment requires all business logic code into separate Java class(es) which could be called from within the servlet to perform specific tasks. All presentation logic is moved to JSP pages. Specifically, this assignment requires you to do the following:

- Make the following modifications on the form:
  - Add text fields on the Student Survey Form to enter city, and state (in lieu of the placeholders using div/span). The user can fills out the city and state field – you do not need to make Ajax call to automatically fill out the city and state fields for a given zip code.
  - You do not need to compute average and maximum of the 10 numbers entered in the Data field in JavaScript thus do not need to keep Average and Maximum fields on the page. You will still keep the Data field on the form.
  - Add an additional field on Student Survey form to enter StudentID. The StudentID uniquely identifies the user.
- Create a StudentDAO class (Note: DAO stands for data access object) that encapsulates code to store and retrieve the Survey data into/from a database. It provides two methods: one to save the Student Survey Form data to a database table and another to retrieve the survey information from the database.
- Develop another Java class called DataProcessor that provides a method to compute the Mean and Standard Deviation using the ten numbers entered in the Data field on the Student Survey Form.
- Develop two acknowledgement JSPs: one to simply thank the user for filling out the form (we call this SimpleAcknowledgement JSP) and the other to announce that the user was a winner of two movie tickets if the mean of numbers was greater than 90 (we call this a WinnerAcknowledgement JSP). Both JSPs will also display the Mean and Standard Deviation computed by the DataProcessor. They also acknowledge that the information

entered on the form was successfully saved to a database table (hoping it was really saved!).

- Develop two JavaBeans: DataBean and StudentBean. The DataBean has two attributes to hold the mean and standard deviation. The StudentBean has attributes that matches most of the Student Survey Form fields, except the Data field.
- This homework involves only one servlet. The servlet acts as a front controller in the MVC implementation and receives and handles all requests from the client, performs business logic via business delegate classes (which may return JavaBean objects), stores the beans into a session or request object, and then forwards the request to appropriate JSP to present the data to the user using a RequestDispatcher object as described below:
    - When a user submits the completed Student Survey Form, the servlet performs two actions: 1) it uses StudentDAO object to store the Student form data to a database table using GMU Oracle database. 2) it calls a method on DataProcessor to compute the mean and standard deviation of the entered 10 numbers. The result of the method call on DataProcessor is a DataBean object with mean and standard deviation. The servlet stores the bean into a session object.
    - If the mean is greater than or equal to 90, the servlet forwards the request to the WinnerAcknlowledgement JSP using RequestDispatcher object. The WinnerAcknowledgement JSP thanks the user for completing the survey, announces that the user is a raffle winner of two movie tickets, and prints mean and standard deviation on the page accessing the data from DataBean.
    - If the mean was less than 90, the servlet forwards request to SimpleAcknowledgement JSP using the RequestDipatcher object. The SimpleAcknowdedgementJSP simply thanks the user for filling out the form and prints mean and standard deviation on the page accessing the data from DataBean.
    - Both pages should have unordered list of hyperlinks of all StudentIDs, which when clicked retrieve the saved form data in the previous steps. The request is forwarded to the same servlet as discussed before, but this time the servlet uses StudentDAO to retrieve the student information from the database table. The return type of this method call is StudentBean object with student data retrieved from the database. The servlet stores this bean to the session object and forwards request to a StudentJSP using RequestDispatcher object to display the student data to the user. The structure of the StudentJSP page for the retrieved data could be similar to the Survey Form in a read only format. If there is no student matching the StudentID, the servlet forwards the request to NoSuchStudentJSP, which informs the user that no student was found with matching StudentID (Note: Since you will be creating your bulleted hyperlinks by reading the data from the database, you will probably not come across situation where no student was found. However, you should still have code to address such situation.)

Submission

The submission for this assignment should be through the blackboard website. I expect a zipped package containing the source files, war file, which could be deployed on a Tomcat server, and any additional packages, scripts, or files that you used. I also require a readme file which contains installation and set up instructions so that the TA and myself can run the assignment.

Submit **all source, and war file, files necessary to run the application** and the installation and execution instructions. Please put all of the files in a **zip** file. If you submit an assignment late the late penalty will apply.

NOTE: A late assignment carries a 10% late penalty for each week it is late. Assignments are NOT accepted after being 2 weeks late.

Make sure your name is on every programming artifact so we know who it belongs to. For every source file, please include comments at the top of the program describing what the program does. This only needs to be 1 or 2 sentences.

Be sure to **test access and functionality** to your submission before the due date.

Grading Rubric

The following areas will be used in the basic grading of the projects:

- Does system meet the functional requirements: 85 Points
- Does the assignment run without errors: 13 Points
- Comments: 2 Points

Instant Point Deductions:

I reserve the right to deduct points instantly for the following reasons:

- The source, or binary, files are not included in the package.
- The readme file is not included in the package.
- The program doesn't run due to errors in the code.
- I spend more than 5 minutes trying to debug the assignment.
- I can't figure out how to use the assignment