SWE 645: Assignment 2

Application overview:

Your second assignment is to create an MVC-based Web application using JSF2 framework. The application allows prospective students to fill out a survey form to provide feedback about their campus visit. It also allows users to view all surveys recorded to date. The application starts with a welcome homepage, which in essence has two links: 1) Student Survey, which allows a prospective student to fill out a survey form, and 2) List All Surveys, which allows a user to view all surveys done to date.

This application could be developed as a dynamic web project in Eclipse and should be deployed as a part of a war file on Tomcat (or Jboss) platform. You should use the Amazon EC2 with a Tomcat or Jboss application server (which you provisioned in previous homework) to deploy your war file on Tomcat or JBoss on Amazon EC2. Also please add a hyperlink of HW2 solution on your class homepage on S3.

The following are more specific requirements of this application.

Requirements:

1. Develop a student survey form (a facelet) to allow a user to enter the survey data. In Particular, the student survey form contains the following:
    a. Text boxes for first name, last name, street address, city, state, zip, telephone number, e-mail, and date of survey, which are required fields. In addition, make sure
        - The name text boxes should contain only alphabets and should not allow more than 15 characters.
        - The address text boxes should contain only appropriate numeric, alphabet, or alphanumeric characters.
        - The length of street address cannot be more than 30.
        - Zipcode should be exactly five digits.
        - Telephone number should have the following pattern: (xxx)-xxx-xxxx
        - The date of survey should have the following pattern: mm/dd/yyyy
        - The Email Address should be valid.

    b. Checkboxes that allow prospective students to indicate what they liked most about the campus. The checkboxes should include: students, location, campus, atmosphere, dorm rooms, and sports.
    c. Radio buttons that allow the prospective students to indicate how they became interested in the university. Options should include: friends, television, Internet, and other.
    d. A dropdown list of options for the user to select the likelihood of him/her recommending this school to other prospective students. The three options of the dropdown list are: Very Likely, Likely, Unlikely.

e. An additional text box called Raffle. The user will be asked to enter at least ten comma separated numbers ranging from 1 through100 in the Raffle field. This information will be used to announce whether the student wins a free movie ticket.
f. A text area for additional comments, and
g. A submit and cancel buttons.

2. Develop two model objects: Student and WinningResult. The Student has properties (instance variables) that matches fields on Student Survey Form. The WinningResult has two properties to hold a mean and standard deviation.

3. Develop a managed bean that contains a reference to Student model object, whose properties are mapped to fields on Survey page.

4. Use implicit/explicit automatic validation approach (as discussed in class slides) to implement validation requirements and provide field specific error checking. For example, if the user submits the survey form without providing data for the required fields, he/she should be directed to the survey input page, with appropriate field-specific error messages.

5. Create a StudentService class that encapsulates business logic to store and read the Survey data to/from a file. It provides methods to save the Student Survey Form data to a file and to retrieve the survey information from the file. (Alternatively, in lieu of using File I/O, you can use a collection placeholder in your managed bean to store the survey data, or use JDBC to store and retrieve survey data into/from a database.) In addition, StudentService provides a method to compute the Mean and Standard Deviation using the ten numbers entered in the Raffle field on the Student Survey Form.

6. Develop two acknowledgement JSF pages: one to simply thank the user for completing the survey (we call this SimpleAcknowledgement.xhtml) and the other to announce that the user was a winner of two movie tickets if the average of numbers entered was greater than 90 (we call this a WinnerAcknowledgement.xhtml). Both JSF pages will also display student name entered on the Survey form as well as the Mean and Standard Deviation computed by the StudentService using the data entered in the Raffle. In addition, both JSF pages also acknowledge that the information entered on the form was successfully saved to a file (hoping it was really saved!).

7. Develop another JSF page called ListSurvey.xhtml which lists all survey records in a tabular format by retrieving the survey records stored in a file to date.

8. Develop a JSF configuration file containing all interactions/navigation of your application.

9. Deploy your war file on Amazon EC2 instance provisioned in previous assignment and add the URL of this homework on your homepage on S3; and provide all this information in readme file as part of your submission.

Above are the core functional requirements. However, feel free to use CSS files for better look and feel of your page and be creative.

Submission

The submission for this assignment should be through the blackboard website. I expect a zipped package containing the source files, war file, 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.

In addition, please add a hyperlink of this homework on your homepage that you developed in first 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:

The following areas will be used in the basic grading of these 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, and instructions are left out.