This assignment involves modifying Assignment 6 to implement the MVC architecture pattern using a RequestDispatcher object. This implementation requires one servlet that acts as a front controller that receives all client requests, performs business logic via business delegate classes, stores JavaBean objects into a session object and then forwards the request to appropriate JSP to present the data to the user. The assignment also requires to move all business logic code from the Servlet class to separate Java classes which could be called from within the servlet to perform specific tasks. All presentation logic is moved to JSP pages. **More specifically,**

- Add an additional field on Student Survey form to enter StudentID. The StudentID uniquely identifies the user.
- Create a StudentDAO class that encapsulates all data access calls. In other words, the StudentDAO class includes all JDBC code of Assignment 6 to store and read the Survey data to/from the database. It provides two methods: one to save the Student Survey Form data to the database and another to retrieve the survey information from the database.
- Develop another Java class called DataProcessor that that provides an interface 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.
- 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 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** 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 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 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.

- The acknowledgement pages should have a placeholder (e.g., a text field) to enter StudentID to retrieve the saved data in the previous step. When a user enters a valid StudentID and submits the request, 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. 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 the entered StudentID is invalid and allow him/her to enter a valid StudentID.