

Noel Miranda

May 8, 2025

Server Side Development

Module 10

### Simple Overview of Code Example

The small dynamic project attached to this submission is the one referenced in the paper. I built it as a mock Bellevue University course survey that allows students to submit feedback for a class they have taken. The application uses JavaServer Faces (JSF) with a request scoped managed bean to handle the form data. The survey collects a student ID, course name, a satisfaction rating from 1 to 10, and written feedback.

For the setup, I used JSF HTML tags to build the user interface, such as `<h:form>`, `<h:inputText>`, and `<h:selectOneMenu>`, which made it easy to bind form fields to the managed bean. The student ID input was tied to `<h:inputText>`, and the rating scale was created using `<h:selectOneMenu>`. I used `<h:commandButton>` to trigger submission and `<h:message>` to display validation feedback below each field when needed. To create a consistent layout, I also used a Facelets template for both the form page and the thank you page. This allowed me to define a common structure and reuse it across views, keeping the design clean and organized.

To ensure clean and proper input, I added Core tag features like `<f:validateRegex>` to enforce a specific format for the student ID and `<f:validateLongRange>` to restrict the rating to a value between 1 and 10. This approach made the form both user friendly and secure without me having to write extra code.

The managed bean is annotated with `@RequestScoped` because the form data only needs to live as long as the request. Once the form is submitted, the bean is discarded, freeing up memory and avoiding any unintended state retention. My plan is to implement a separate model

class in the future that will handle the business logic for submitting the survey data to a database. Once that is in place, the managed bean will act as a bridge between the user interface and this backend logic, making it easier to expand the application while keeping concerns separated.

Overall, the JSF application goes through the JSF lifecycle to collect user feedback through the simple form. It validates the data before processing it and, if all goes well, displays a confirmation page with a thank you message. If validation fails, the JSF lifecycle is interrupted to display error messages below the form's fields.

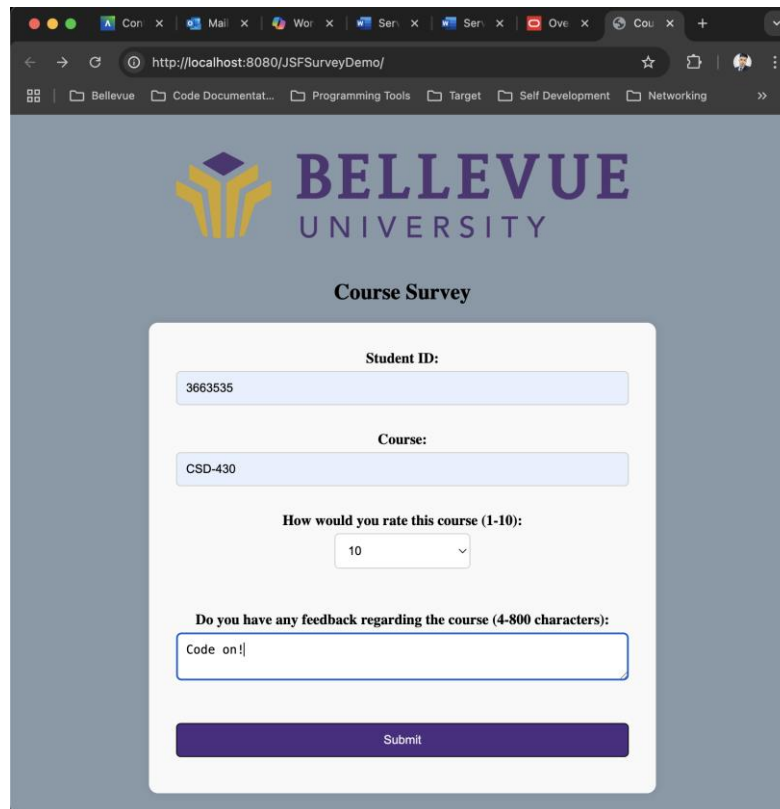
Images Below of the Code Example During Test & Executing Successfully:

A screenshot of a web browser displaying the Bellevue University Course Survey form. The browser's address bar shows the URL `http://localhost:8080/JSFSurveyDemo/`. The page features the Bellevue University logo at the top, followed by the title "Course Survey". The form is a white box with a light gray border, containing the following fields and labels:

- Student ID:** A text input field.
- Course:** A text input field.
- How would you rate this course (1-10):** A dropdown menu with the text "-- Select Rating --".
- Do you have any feedback regarding the course (4-800 characters):** A text area.
- Submit**: A purple button at the bottom of the form.

A screenshot of the same Bellevue University Course Survey form, but with validation errors displayed in red text. The browser's address bar shows the URL `http://localhost:8080/JSFSurveyDemo/index.xhtml`. The form fields and labels are the same as in the previous image, but with the following error messages:

- Student ID:** Below the input field, the text "Student ID is required." is displayed.
- Course:** Below the input field, the text "Course is required." is displayed.
- How would you rate this course (1-10):** Below the dropdown menu, the text "j\_idt13:rating: Validation Error: Specified attribute is not between the expected values of 1 and 10." is displayed.
- Submit**: The purple button remains at the bottom.



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/JSFSurveyDemo/`. The browser's tab bar shows several open tabs including 'Cor...', 'Mail', 'Wor...', 'Ser...', 'Ser...', 'Ove...', and 'Cou...'. The browser's bookmark bar contains 'Bellevue', 'Code Documentat...', 'Programming Tools', 'Target', 'Self Development', and 'Networking'. The main content area features the Bellevue University logo (a stylized yellow and purple icon) and the text 'BELLEVUE UNIVERSITY'. Below this is the heading 'Course Survey'. The form itself is a white box with a light blue border. It contains the following fields: 'Student ID:' with the value '3663535', 'Course:' with the value 'CSD-430', 'How would you rate this course (1-10):' with a dropdown menu showing '10', and 'Do you have any feedback regarding the course (4-800 characters):' with a text area containing 'Code on!!'. A purple 'Submit' button is at the bottom of the form.

**BELLEVUE UNIVERSITY**

**Course Survey**

**Student ID:**

3663535

**Course:**

CSD-430

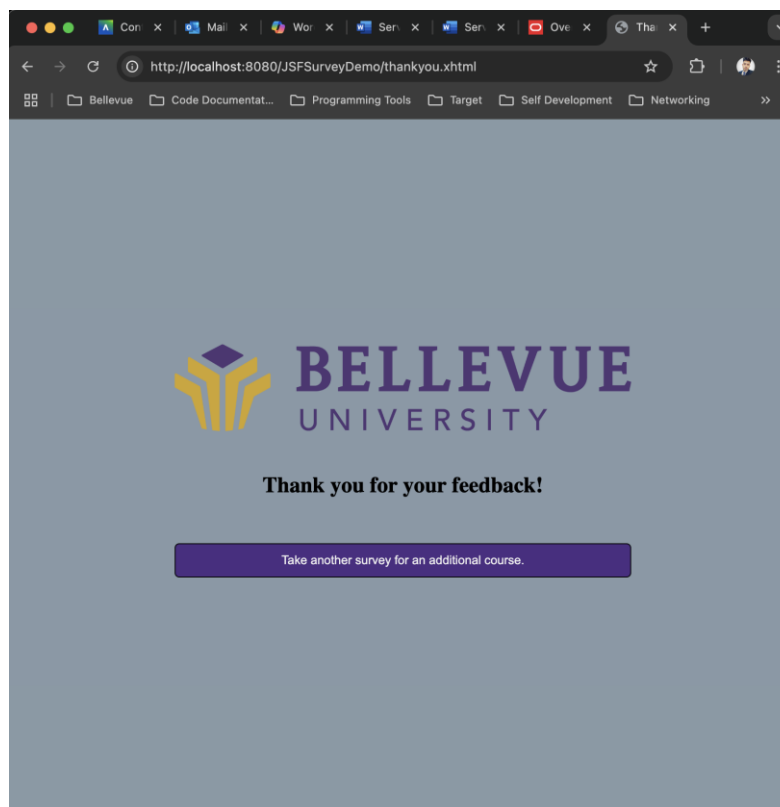
**How would you rate this course (1-10):**

10

**Do you have any feedback regarding the course (4-800 characters):**

Code on!!

**Submit**



The screenshot shows a web browser window with the address bar displaying `http://localhost:8080/JSFSurveyDemo/thankyou.xhtml`. The browser's tab bar shows several open tabs including 'Cor...', 'Mail', 'Wor...', 'Ser...', 'Ser...', 'Ove...', and 'Tha...'. The browser's bookmark bar contains 'Bellevue', 'Code Documentat...', 'Programming Tools', 'Target', 'Self Development', and 'Networking'. The main content area features the Bellevue University logo (a stylized yellow and purple icon) and the text 'BELLEVUE UNIVERSITY'. Below this is the heading 'Thank you for your feedback!'. At the bottom, there is a purple button with the text 'Take another survey for an additional course.'.

**BELLEVUE UNIVERSITY**

**Thank you for your feedback!**

**Take another survey for an additional course.**