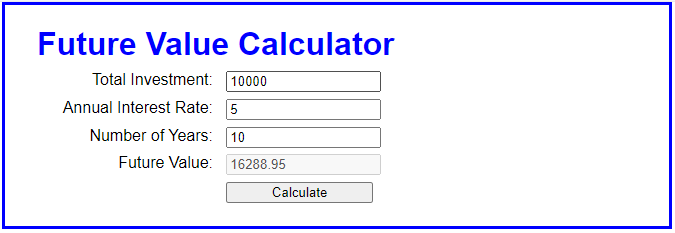
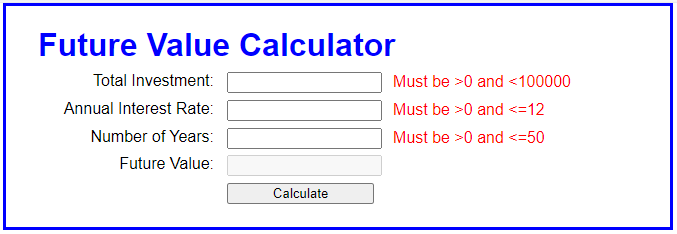
Lab 3.4 Future Value application

In this application, you’ll build a new version of the Future Value application. Its user interface will look as follows:



Print the error messages in the <span> tags as follows:



**Open and review the starting files**

1. Open the files in this folder:

**\Future Value\**

1. Run the HTML file to see that it provides the user interface, but nothing works. Then, review the HTML code, and note that the span elements contain non-breaking spaces (&nbsp;). That way, the text nodes for these elements will be added to the DOM.
2. Review the JavaScript code. Note that it contains just the $ function.

**Create a function for the future value calculation**

1. Create a new function called calculateFV. It should have 3 parameters that receive the users entries: investment amount, interest rate and number of years. It should calculate the future value based on these parameter values, round the result to 2 decimal places, and return the rounded result.

**Create the event handler for the click event of the calculate button**

1. Create a function expression named processEntries that gets the user entries with no data validation. Use strict mode, and start by declaring the variables that will hold the users entries and assigning the users entries to these variables.
2. Code a statement that calls the calculateFV function and stores the result that’s returned in the fourth text box.
3. Create an event handler for the window load event that attaches the processEntries function to the click event of the calculate button. This handler should also move the focus to the first text box.
4. Test the application with valid entries and debug until it works correctly.

**Add data validation with error messages to the right of the entries**

1. Declare any variables that you are going to need for data validation right after the other declarations. For instance, you may want to use an isValid variable, depending on how you code the validation routines.
2. Add data validation that tests whether the first entry is a valid number that’s greater than zero and less than or equal to 100000, and display an appropriate error message in the span element after the investment text box if it isn’t.
3. If the entry is valid, the else clause should issue the statement that you coded in step 6. But it should also set the contents of the span element for the entry to an empty string (“”).
4. Add similar data validation to the next 2 entries.