ASP.NET Core Assignment - Library Due Date Tracker Day 3

Deadline: Monday, November 9th 2020, 09:00 AM

f[GitHub Classroom Link]

Introduction

This assignment is meant to challenge your mastery of ASP.NET Web Application (Model - View - Controller) and how well you are able to use MVC to create a CRUD application. Your goal in this assignment is to create a tool that will help you keep track of all the books you have checked out of the library. This is a cumulative activity. Use your code from **ASP.NET Core Assignment - Library Due Date Tracker Day 2** as a starting point.

Requirements

- ❏ Modify "Borrow" (Model):
  - ❏ Add a property "ExtensionCount" - int(10), not nullable.
    - Update your seed data for this table to include values for this field.
  - ❏ Add a migration.
  - ❏ Update the database.
- ❏ Modify "List" (View / Action):
  - ❏ Create a form with a checkbox "Filter to Overdue".
    - When the page loads with the checkbox checked (query string parameter), call the "GetOverdueBooks()" method instead of the "GetBooks()" method.
- ❏ Modify "Details" (View / Action):
  - ❏ Add a "Number of Extensions" line / output.
- ❏ Add the following business logic to the controllers:
  - ❏ General Validation:
    - Trimmed all data prior to processing.
    - All comparison validation must be case insensitive.
    - String data cannot exceed its database size.
    - NOT NULL fields must have values that are not whitespace.
    - All numeric/date fields must successfully parse.
    - Primary keys on 'ByID' methods must exist.
    - Reference IDs (foreign keys) must exist in their respective tables.
  - ❏ Library Business Logic:
    - "CheckedOutDate" cannot be prior to "PublicationDate".
    - "ReturnedDate" cannot be prior to "CheckedOutDate".
    - "PublishedDate" cannot be in the future.
    - An extension must actually extend the due date in order to be valid.
    - Overdue books cannot be extended.
    - Books cannot be extended more than 3 times.
    - Book titles must be unique for that author.
- ❏ Display itemized errors on all appropriate "Book" view pages.

Challenges (See Rubric for Details)

- ❑ Make it look nice with CSS.
- ❑ Modify “List” (View) to show the user how many days a book is overdue, and make the text dark red.
- ❑ Add an “Archived” flag to “Book” that will become the new method for “DeleteBookByID”.
    - Set the flag to true when a book is deleted.
    - Don’t allow a book that isn’t returned to be archived.
    - Don’t show the book on “List” unless archived books are being shown.
    - Don’t allow any borrows to take place for the book.
- ❑ Modify the checkbox on “List” (View) to be a dropdown with multiple types of filters:
    - All Books
    - In-Stock Books
    - Lent Books
    - Overdue Books (Included in Lent Books)
    - Archived Books
- ❑ Create a “Report” Action / View that will provide some summaries about the data.
    - Determine which author’s books have the longest total checked-out time.
        - This should work with books that haven’t been returned, as well as on books that have been returned.
- ❑ Have an unexpected feature.

Hints

- General Hints:
    - Focus on the requirements first, challenges are extra!
    - This kind of project has been done by many others in the past! Don’t hesitate to use your google-fu skills if you don’t know how to implement certain features!
    - Please include source citations in your code and README.md
- Day 3 Hints:
    - All changes outlined in day 3 requirements should be done server side before passing the information off to the database; existing book records should be edited by ID
    - Draw up existing records by ID in order to match information against one another (example: recall CheckoutDate from the database to match it against ReturnDate)

Screenshots

List

Filter to Overdue ☑ Filter

ID	Title	Author	Publication Date	Checked Out Date	Due Date	Returned Date
-7	Hamlet	William Shakespeare	January 1, 1600	September 23, 2020	October 7, 2020	Not Returned
-3	How the Grinch Stole Christmas	Dr. Seuss	October 12, 1957	December 25, 2019	January 8, 2020	Not Returned
1	It Came from Beneath the Sink!	R.L. Stein	April 1, 1995	September 7, 2020	September 21, 2020	Not Returned
2	Test 123	H.P. Lovecraft	January 1, 2020	February 2, 2020	September 30, 2020	Not Returned

List

Filter to Overdue ☐ Filter

ID	Title	Author	Publication Date	Checked Out Date	Due Date	Returned Date
-7	Hamlet	William Shakespeare	January 1, 1600	September 23, 2020	October 7, 2020	Not Returned
-6	Animal Farm	George Orwell	August 17, 1945	July 2, 2020	August 6, 2020	August 1, 2020
-5	The Call of Cthulhu	H.P. Lovecraft	February 1, 1928	April 22, 2020	May 20, 2020	May 18, 2020
-4	Nineteen Eighty-Four	George Orwell	June 8, 1949	November 17, 2018	December 1, 2018	November 19, 2018
-3	How the Grinch Stole Christmas	Dr. Seuss	October 12, 1957	December 25, 2019	January 8, 2020	Not Returned
-2	The Cat in the Hat	Dr. Seuss	March 12, 1957	December 25, 2019	January 15, 2020	January 12, 2020
-1	Green Eggs and Ham	Dr. Seuss	August 12, 1960	December 25, 2019	October 1, 2020	September 24, 2020
1	It Came from Beneath the Sink!	R.L. Stein	April 1, 1995	September 7, 2020	September 21, 2020	Not Returned
2	Test 123	H.P. Lovecraft	January 1, 2020	February 2, 2020	September 30, 2020	Not Returned
4	New Book	H.P. Lovecraft	January 3, 2020	September 24, 2020	October 1, 2020	September 24, 2020

Details - 1

That book is overdue, it cannot be extended. Please see the librarian.

You checked out It Came from Beneath the Sink! on September 7, 2020, and it was due on September 21, 2020.

Book Details

Title: It Came from Beneath the Sink!

Author: R.L. Stein

Publication Date: April 1, 1995

Library Details

Checked Out Date: September 7, 2020

Due Date: September 21, 2020

Extensions Granted: 0

Returned Date: Not Returned

Return 1 - It Came from Beneath the Sink! | Extend Deadline of 1 - It Came from Beneath the Sink! | Delete 1 - It Came from Beneath the Sink!

Citation Guide for Borrowed Code

Whenever you borrow code, the following information must be included:

- Comments to indicate both where the borrowed code begins and ends.
- A source linking to where you found the code (URL, book, example, etc.).
- Your reason for adding the code to your assignment or project instead of writing it out yourself.
- Explain to us how the code is supposed to work, include links to documentation and articles you read to help you understand.
- A small demonstration to prove you understand how the code works.

```
const inputArr = [5,1,3,4,2];

/*Borrowed code for bubbleSort starts*/
let bubbleSort = (inputArr) => {
    let len = inputArr.length;
    for (let i = 0; i < len; i++) {
        for (let j = 0; j < len; j++) {
            if (inputArr[j] > inputArr[j + 1]) {
                let tmp = inputArr[j];
                inputArr[j] = inputArr[j + 1];
                inputArr[j + 1] = tmp;
            }
        }
    }
    return inputArr;
};

/*Borrowed code from bubbleSort ends*/

//Source: bubbleSort function obtained from https://medium.com/javascript-algorithms/javascript-algorithms-bubble-sort-3d27f285c3b2
//Reason to add: implementing bubble sort can be tedious and bug prone, it would be better to use a proven version than to write my own
//How it works: I read the following article to understand how bubble sorts work (http://www.pkirs.utep.edu/CIS3355/Tutorials/chapter9/tutorial9A/bubblesort.htm)
//Demonstration of understanding:
//Example array: [3,1,2]
//Step 1: Compare 3 and 1. Since 1 is smaller, swap places.
//Array: [1,3,2]
//Step 2: Compare 3 and 2. Since 2 is smaller, swap places.
//Array: [1,2,3]
//Step 3: Compare 1 and 2. No need to swap.
//Array: [1,2,3]
//Step 4: Compare 2 and 3. No need to swap.
//Array: [1,2,3]
//Function complete.
console.log(bubbleSort(inputArr));
```