IT 4203 Individual Project Milestone #1: Google Books API

Fall 2016 – updated on 9/27

Due Date: See D2L

Instruction

We will continue the project and use real world web API in this milestone.

Requirements:

- Use Google Books API
- Create the following three pages. Provide a main menu with two items: "home" (book search) and "my bookshelf"
- Book search page: for users to search books by key terms.
    - A textbox to accept user input of search terms.
    - After the search button is clicked, dynamically construct the service request URL and call the service.
    - Process the service response (JSON format) and display search results (no need to do AJAX at this time, but you can if you know how to use JSON-P and those AJAX functions).
    - Display the first 100 results (at most) with paging (you can decide the number of results per page and number of pages).
    - For each book in the result, display book title and a smaller cover image. Then create a link on the title which links to the second page to display detailed information of that book.
- Book details page: to display all information about a single book.
    - Accept a book id parameter from the URL and dynamically constructed the API URL.
    - Display as many details as you can about a single book (examine the JSON content), such as title, publisher, authors, description, image, price, etc.
    - Organize the information in a decent clean style/layout.
- Your public bookshelf page: you should display books from one of your public bookshelves.
    - Create a public bookshelf in your library at the Google Books website (https://books.google.com) and add some your preferred books first. You will do this manually through the Google Books website in this milestone. We will add the functionality of adding books through API and AJAX in the next milestone.
    - Display your books in this bookshelf, with each linked to your book details page.

Other requirements

- Please create a separate folder and style for your milestone #2. Make it independent and look like a web app by its own. You will design your own style but keep them clean and organized. Here is a simple and clean design example: http://it-ebooks.org/search/?q=modern+web
- Images and URLs should be displayed or function correctly.
- Do not use any other library or framework other than jQuery.
- All web pages should be able to be accessed from your public website lively.

[Challenge/Bonus – 1 point]

Embed book content preview to the book details page, using the Book Preview API (https://developers.google.com/books/docs/viewer/developers_guide)

Submission

Submit the following to D2L:

1. Compress all source code files into a .zip file.
2. A report in Microsoft Word (or PDF), including:
   - The URL to your project and/or the milestone web pages.
   - Screenshots with explanations. Take a screenshot of each webpage displayed in the browser (only the top part of the page if the page is too long). Make sure your screenshots are clearly shown and without cropping.

Grading guide

Your grade is determined on:

- How much you have satisfied the requirements specified in the instruction, and your creativity
- Web site/page design quality
- Well-formed and clean code. You should not use any wizard to generate the code, nor should you copy and paste codes from other sources without cleaning it.
- Following the submission requirements

Rubric:

Score	Summary	Rating Description
10	Outstanding work; beyond expectation.	Correctly read content from all APIs. Satisfy all requirements. Excellent web site/page design. Well-organized files and code; easy-to-read and clean code.
8-9	Good work; meet expectations	Correctly read/display content from all APIs, with possible minor mistakes or missing elements. Mostly satisfy requirements but with minor issues. Good web site/page design (neat and clean). Well-organized files and code; easy-to-read and clean code.
6-7	Adequate work;	Correctly read/display content from all APIs, with quite some mistakes or missing items. Missing one or two major requirements. Too simple web page design (missing image or links, or incorrectly displayed). Have some problems of organizing code; difficult to read. Missing some required submission items.
<6	Lack of effort.	Failed to read content correctly from any API. Poor web page design (unorganized HTML). Poor coding practices. Live website does not work. Fail to follow submission requirements.