WhatABook Business Requirements

Scenario

WhatABook, a small used book store has decided to expand their presence by offering customers a solution they can install on their personal computers. WhatABook would like to establish an application that allows customers to browse their in-store book listing, allow customers to add books to a wishlist, allow customers to view their wishlist, show a listing of books by genre, show a listing of books by author, search for book by title, and search for book by bookId. The interface must be as simplistic as possible, because most of WhatABook's customers are middle age users with minimal computer experience. WhatABook has given you three (3) weeks to complete the project.

Business Rules

1. WhatABook has many BOOK(s).
2. a CUSTOMER can add many BOOK(s) to their WISHLIST
3. a CUSTOMER can have many WISHLIST BOOK(s).
4. WhatABook has many CUSTOMER(s)

Data Model Fields

title	genre	author	bookId	firstName
lastName	customerId			

Database Guidance

1. You do not need a collection for WhatABook because the database will only support one company and it is inferred that BOOK(s) are owned by WhatABook.
2. There could be three (3) collections (books, customers, and wishlistitems).
   a. If you choose this route, lookup ID's should be added to each of the collections to make the data joinable; for example, consider how you leveraged the $lookup aggregate operation in assignment 6.2. In this scenario, customerId and bookId should be added to the wishlistitems collection. This way you can use the $lookup

operation to join the wishlistitems and books collections to show a list of wishlistitems by customerId.

3. There could be two (2) collections (books and customers).
    a. If you choose this route you will need to leverage embedded documents. That is, wishlistitems would be embedded documents under the customer document.

Requirements

Technical Design Document (TDD):

1. Identify the team captain, team name, mascot, and provide a short bio of each team member.
2. Identify the goals of the website.
3. Identify the core users of the website.
4. Create at least three (3) detailed personas to represent the core users of the website.
5. Create five user stories per persona using standard story writing format (there should be a total of fifteen (15) user stories).
6. Add a column next to each story for story points and rank each task using the Fibonacci sequence.

Special note. The name and mascot you choose must be explained in the document (i.e., why was the team name/mascot selected and what is its meaning). Also, you are not building an actual website for this group project. Instead, your team is building the MongoDB database scripts, queries, and a console application through Python.

Database Diagrams:

1. ORD
2. NoSQL Data Structure

Special note. The ORD and NoSQL Data Structure will be added to the TDD.

User Interface Prototype:

1. Functional prototype for the proposed application (target websites)

Special note. The functional prototype will be added to the TDD.

Required Queries:

Special note: The queries you write in this project should be saved to a file named <groupName>-<projectName>.js and added to your GitHub repository under a folder called whatabook. Use standard programming comments above each query to explain what the query is doing. In addition, do not forget code attribution

1. Display a list of books.
2. Display a list of books by genre.
3. Display a list of books by author.
4. Display a book by bookId.
5. Display a wishlist by customerId.
6. Add books to a customer's wishlist.
7. Remove book from a customer's wishlist.

Console Application Requirements:

1. Connect to your MongoDB database.
2. Display a list of books. Format the output so it is easy to read.
3. Display a list of books by Genre. For this requirement, supply the user with a list of genre choices and display the appropriate books based on their selection.
4. Display a customers wishlist by customerId. For this requirement, prompt the user to enter a customerId (c1007, c1008, or c1009) and display the appropriate wishlist.
5. Add basic error handling to account for an invalid customerId (hint: use an if…else or switch statement).