**Design Document: Bazar.com - Online Book Store**

**1. Introduction and Overview:**

Bazar.com is an online book store that aims to provide customers with a selection of books and a seamless purchasing experience. This design document outlines the system architecture, functionality, and implementation details for the Bazar.com application.

**2. System Architecture:**

The Bazar.com application follows a two-tier web architecture, consisting of a front-end server and two microservices: the catalog server and the order server. The front-end server handles user requests and communicates with the catalog server and order server over HTTP REST APIs.

**3. Functional Requirements:**

* Users can search for books based on topic.
* Users can view detailed information about a book, including stock availability and cost.
* Users can purchase books by providing the item number.
* The catalog server should support query-by-subject and query-by-item operations.
* The order server should handle customer purchase requests.

**4. Data Model:**

The data model consists of three main entities: Book, Order and catalog.

Catalog:

* id: Unique identifier for each catalog.
* name: Name of the catalog.

Book:

* id: Unique identifier for each book.
* name: Title of the book.
* count: Number of items available in stock.
* price: Cost of the book.
* catalog\_id: Foreign key referencing the catalog the book belongs to.

Order:

* id: Unique identifier for each order.
* book\_id: Foreign key referencing the book that was purchased.
* purchase\_date: Date when the purchase was made.
* count: Number of books purchased.

**5. Component Design:**

Front-end Server: Handles user requests, interacts with the catalog server and order server APIs, and returns responses to users.

Catalog Server: Manages the book catalog, supports query operations, and provides book information to the front-end server.

Order Server: Handles customer purchase requests, verifies book availability with the catalog server, and processes the purchase.

**6. Implementation Plan:**

* Milestone 1: Set up the project repository, create the project structure, and implement the basic server components.
* Milestone 2: Implement the catalog server functionality, including book catalog management and query operations.
* Milestone 3: Develop the order server to handle purchase requests and integrate it with the catalog server.
* Milestone 4: Implement the front-end server, including user request handling and communication with the catalog and order servers.
* Milestone 5: Perform testing and debugging, ensuring the system functions correctly and meets the requirements.
* Milestone 6: Finalize documentation, review the design, and prepare for deployment.

**7.Extentions:**

- User Authentication and Authorization:

* Front-end Server:
  + Sends authentication requests to the authentication microservice for user login and session management.
  + Sends purchase queries to the order server when a user wants to make a purchase.
  + Sends search and information queries to the catalog server to retrieve book details and perform searches.
* Authentication Microservice:
  + Handles user authentication and authorization.
  + Manages user account creation, login, and session management.
  + Responds to authentication requests from the front-end server to verify user credentials and provide authorization.
* Order Server:
  + Receives purchase queries from the front-end server when a user wants to make a purchase.
  + Sends authentication requests to the authentication microservice to verify user authorization before processing the purchase.
  + Processes the purchase query, including verifying the availability of books and updating order-related information.
  + Communicates with the catalog server to update book inventory after a successful purchase.
* Catalog Server:
  + Handles search queries from the front-end server to retrieve book information based on user search criteria.
  + Provides information queries from the front-end server to fetch specific details about books.
  + Receives update queries from the order server to adjust book inventory after a purchase.

Authentication  

Front end

Search

Authentication

Authentication request 

purchase 

Info

Authentication request 

Update 

Catalog server

Order server

**To run it**, follow the instructions in the "docker\_setup" file and use Postman to test the application using the provided URLs.

<http://0.0.0.0:4000/> => catalog server

<http://0.0.0.0:5000/> => order server

[http://0.0.0.0:3000/](http://0.0.0.0:3000/info/1) => front server

e.g:  
<http://0.0.0.0:3000/info/1> to get information the book with id 1