Design for Bazar.com: A Multi-tier Online Book Store

Overview:

Bazar.com is a multi-tier online book store implemented using a microservices architecture.

We used the web framework Flask for python language, It contains 3 servers: Front-end server , Catalog server and Order server.

We used Docker to run them.

1. Front-end server :

User interface for the online book store supports three operations: search, info, and purchase .

These operations trigger corresponding requests to the catalog and order services.

2. Catalog server :

Supports query (by item number & topic name) and update(stock or cost) operations.

3. Order server :

Supports a single operation: purchase from the Frontend Server.

How It Works ?

Front-end Server:

- Users interact with the front-end server through the RESTful API, making requests for search, info, or purchase.
- Search and info operations trigger queries to the catalog server, and purchase operation triggers request to the order server.

Catalog Server:

- queries for book information and updates to the catalog.
- Supports both query-by-subject and query-by-item operations.

Order Server:

- Processes purchase requests by verifying stock availability through the catalog server.
- Decreases the stock if the item is available.

Design Trade-offs:

- Synchronization and Consistency:
  Take advantage of the concurrency support provided by modern web frameworks instead of implementing low-level thread code to handle concurrent requests.

- Web Framework (Flask):
  Tradeoff: Choosing Flask as the web framework for implementing microservices.
  Flask is a lightweight and easy-to-use web framework, making it well-suited for small-scale applications and microservices.

- Using the RESTful API:
  Trade-off: Implementing the system as a RESTful API.
  RESTful APIs are widely adopted, easy to understand, and work well with microservice type communication. RESTful APIs were chosen for their simplicity and suitability for this project.

Improvements and Extensions:

- ✓ Implement secure communication using HTTPS encrypting communication between components enhances security and protects sensitive information such purchase details.

- ✓ User Authentication and Authorization:
  Implement user authentication and authorization mechanisms. Introducing user accounts and authentication enhances security .