An-Najah National University

Department of Computer Engineering

Distributed Operation Systems - 10636456

Lab #1 - Report

Project – part 1: Bazar.com

A Multi-tier Online Book Store

Students:

Raghad Sabri 11924224

Yaqout Salameh 11924020

In this project, we did everything together but we made Zoom and pushed the project to GitHub from one device most of the time.

In this project, we used Spark Java language (Spark Framework - Create web applications in Java rapidly. Spark is a micro web framework that lets you focus on writing your code, not boilerplate code) to implement the different APIs that we needed to achieve what we asked to do in this project with the helped with SQLite database (SQLite is an open-source, zero-configuration, self-contained, stand-alone, transaction relational database engine designed to be embedded into an application) to handle the shared data in concurrent way between the different container.

We are asked to design Bazar.com - the World's smallest bookstore.Bazar.com carries only two types of books each type contains only two different books.

We have two tiers, the frontend and the backend, in the frontend we made a user-friendly design accepted to control the backend servers (the catalog and the order servers).

The catalog server is responsible for accepting the search and the info APIs, the search API is used to get the ID and the title of the books depending on the specific book topic that I specify in the URL get API, and the info API is used to get the id, topic, title, quantity and the price of the books depend on the specific book id that I specify in the URL get API (we use the get API because we need to return values). In addition, the put API is responsible for decreasing the books' quantity that we ordered by specifying the book ID (we use the put API because we need to update values). All this information had been handled by using the catalog database which contains the book table.

The order server is responsible for making orders, in this server we make a post API this API is used to make orders with the book ID (which I specify in the URL post API) and the order ID (we use the post API because we need to create a new order), but before that, we need to check the required book quantity if it is zero so we can't make an order otherwise we can make an order this done by using the get API since we make an HTTP connection and to send the get, and the put request, the get request send to the catalog server to fetch the information for the required book id by info API, if the quantity is not zero we continue to the put request which is sent to the catalog server to decrease the books' quantity by one using the put

API. All the orders are added to the orders table in the orderserver database.

In the frontend, we make a simple and user-friendly design first we choose one of the features search, info, or order, if we choose search we need to send the topic of the book, if we choose info we need to send the ID of the book, and if we choose order we need to send the id of the book we need to order. The system will reject any invalid data like choose wrong type of data, or unavailable book choice (the book does not exist or empty stock), etc…

To improve our program, we can add additional delete, get, post, and put APIs in the catalog server, the post API to add new books to the book table in the catalog database, the put API to update any value in the book table after specifying the book(or books) like updating the ID, topic, title, quantity, and price, and the delete API to delete any book(or books) in the book table after specifying the book(or books), the get API to get any book(or book) in the book table after specifying the book(or books) depend on the title, quantity, or price.

We can add additional delete, get, and put APIs in the order server, the put API updates any value in the orders table in orderserver database after specifying the order(or orders) like updating the order ID, book ID, and the delete API to delete any order(or orders) in the orders table after specifying the order(or orders), the get API to get any order(or orders) in the book table after specifying the order(or orders). But we need to be careful that any change in order we need to make changes in books’ quantity in the book table in the catalog database.

GitHub Repository:

https://github.com/YaqoutS/dos-project_online-book-store/tree/catalog-server