DISTRIBUTED AND OPERATING SYSTEMS

Bazar.com: A Multi-tier Online Book Store

Wafaa Daas – 11715407
Israa Ahmad - 11715700

The program was built using multi-tiers architecture which is the front end tier and the back end tier (the server which is separated into two tiers).The front end tier which is the client is the first server that sends the requests to the back end tier. The other part which is the back end is separated into two servers - The catalog server and the order server).

The Catalog server is responsible for storing data related to the books in the store and it does the requests of fetching the information of books and updating some of this information, while the order server is mainly responsible for the purchase operation.

The program is built using flask with SQLite database, it is a web REST API which contains micro-serves in each server.

Every server remains on a different machine (the front server was in Windows and the other two servers were in Ubuntu) and they communicate with each other using the network using the HTTP protocol.

To implement the program we use two virtual machines and real OS, we put the first tier which is the front end tier. And the other two were in Ubuntu. Then we give each machine static IP (using bridge network) in order to set up the communication between the machines. Then run the three servers together and use Postman to send requests to test the required functions.

The user will send the request to the front end tier using Postman and the server will send the request to the back end servers in order to achieve the required functions which are:

- Information (book_id): this request will be like the form below "http://192.168.1.30:5000/information/<book_id> “this request will reach the front end tier then it will be sent to the Catalog server which will get the whole information of the required book and send it back to the client as a JSON file then.

- Search (topic): this request will be like the form below "http://192.168.1.30:5000/search/<book_id> “this request will reach the front end tier then it will be sent to the Catalog server. Then the response will be a JSON file containing the books related to the sent topic.

- Purchase (book_id): this request will be like the form below "http://192.168.1.30:5000/purchase/<book_id> “this request will reach the front end tier then it will be sent to the Order server. Then the Order server will send a request to the Catalog server to check if there are enough books of the required book. If there is then the server will decrease the amount of that book and return an approval response to the Order server and that server will send a response that contains an approval message. If not then it will return a response that tells there are not enough books.

Also, there are other functions such as decrement the number of specific books, incrementing the number of specific books, updating the price of a specific book, and getting the information of all available books.

As an improvement, we can move purchase operation to the Catalog server instead of the Order server, if we do that then the Catalog server will start by checking if there is enough amount of books; if there is then send a request to the Order server; if not it will send the response back to the front without any requests to Order server.

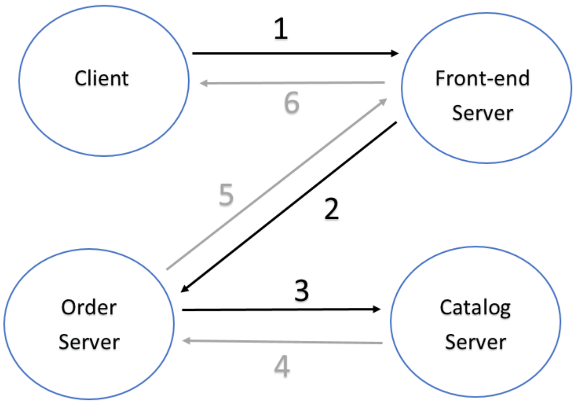

Figure 1: Before

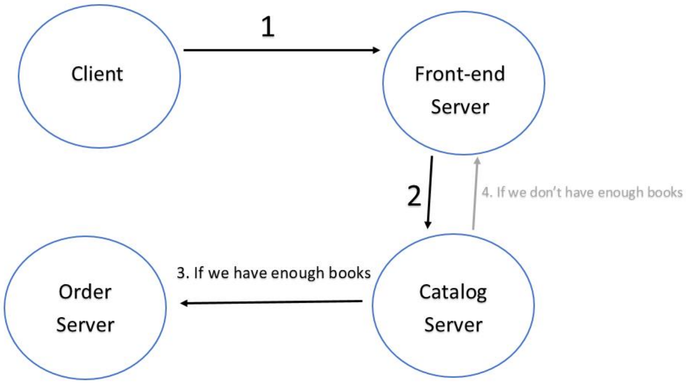

Figure 2: After